systems

MDPI

Article

Using Dual Attention BiLSTM to Predict Vehicle Lane Changing Maneuvers on Highway Dataset

Farzeen Ashfaq [1], Rania M. Ghoniem [2,*], N. Z. Jhanjhi [1], Navid Ali Khan [1] and Abeer D. Algarni [2]

[1] School of Computer Science, SCS, Taylor's University, Subang Jaya 47500, Malaysia; noorzaman.jhanjhi@taylors.edu.my (N.Z.J.)
[2] Department of Information Technology, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia
* Correspondence: rmghoniem@pnu.edu.sa

Abstract: In this research, we address the problem of accurately predicting lane-change maneuvers on highways. Lane-change maneuvers are a critical aspect of highway safety and traffic flow, and the accurate prediction of these maneuvers can have significant implications for both. However, current methods for lane-change prediction are limited in their ability to handle naturalistic driving scenarios and often require large amounts of labeled data. Our proposed model uses a bidirectional long short-term memory (BiLSTM) network to analyze naturalistic vehicle trajectories recorded from multiple sensors on German highways. To handle the temporal aspect of vehicle behavior, we utilized a sliding window approach, considering both the preceding and following vehicles' trajectories. To tackle class imbalances in the data, we introduced rolling mean computed weights. Our extensive feature engineering process resulted in a comprehensive feature set to train the model. The proposed model fills the gap in the state-of-the-art lane change prediction methods and can be applied in advanced driver assistance systems (ADAS) and autonomous driving systems. Our results show that the BiLSTM-based approach with the sliding window technique effectively predicts lane changes with 86% test accuracy and a test loss of 0.325 by considering the context of the input data in both the past and future. The F1 score of 0.52, precision of 0.41, recall of 0.75, accuracy of 0.86, and AUC of 0.81 also demonstrate the model's high ability to distinguish between the two target classes. Furthermore, the model achieved an accuracy of 83.65% with a loss value of 0.3306 on the other half of the data samples, and the validation accuracy was observed to improve over these epochs, reaching the highest validation accuracy of 92.53%. The F1 score of 0.51, precision of 0.36, recall of 0.89, accuracy of 0.82, and AUC of 0.85 on this data sample also demonstrate the model's strong ability to identify both positive and negative classes. Overall, our proposed approach outperforms existing methods and can significantly contribute to improving highway safety and traffic flow.

Keywords: lane-change prediction; BiLSTM; naturalistic vehicle trajectories

Citation: Ashfaq, F.; Ghoniem, R.M.; Jhanjhi, N.Z.; Khan, N.A.; Algarni, A.D. Using Dual Attention BiLSTM to Predict Vehicle Lane Changing Maneuvers on Highway Dataset. *Systems* **2023**, *11*, 196. https://doi.org/10.3390/systems11040196

Academic Editor: William T. Scherer

Received: 5 March 2023
Revised: 1 April 2023
Accepted: 3 April 2023
Published: 14 April 2023

1. Introduction

Lane-changing maneuvers refer to the action of a car transferring from a lane to another on a multi-lane roadway. This maneuver is typically performed by vehicles when they wish to pass another vehicle, change direction, or exit the roadway. Lane changes can pose a challenge for drivers as they require them to assess the traffic flow in neighboring lanes and determine the safety of switching lanes. This task is further complicated by the need for precise control of both the vehicle's longitudinal and lateral movements. In the United States alone, approximately 539,000 two-lane traffic incidents occur annually [1]. Factors that can affect the safety of a lane change include the speed and distance of other vehicles, the visibility of the road ahead, and the presence of any obstacles or hazards. To make a lane change, the driver has to signal their intent, check mirrors, check blind spots, and then execute the maneuver. The prediction of lane-change maneuvers is a

critical task in the field of advanced driver-assistance systems (ADAS) [2,3], autonomous driving [4–7], and intelligent transportation systems (ITS) [8–11]. Accurately predicting when a vehicle will change lanes can have significant implications for highway safety, traffic flow, and traffic management as depicted in Figure 1.

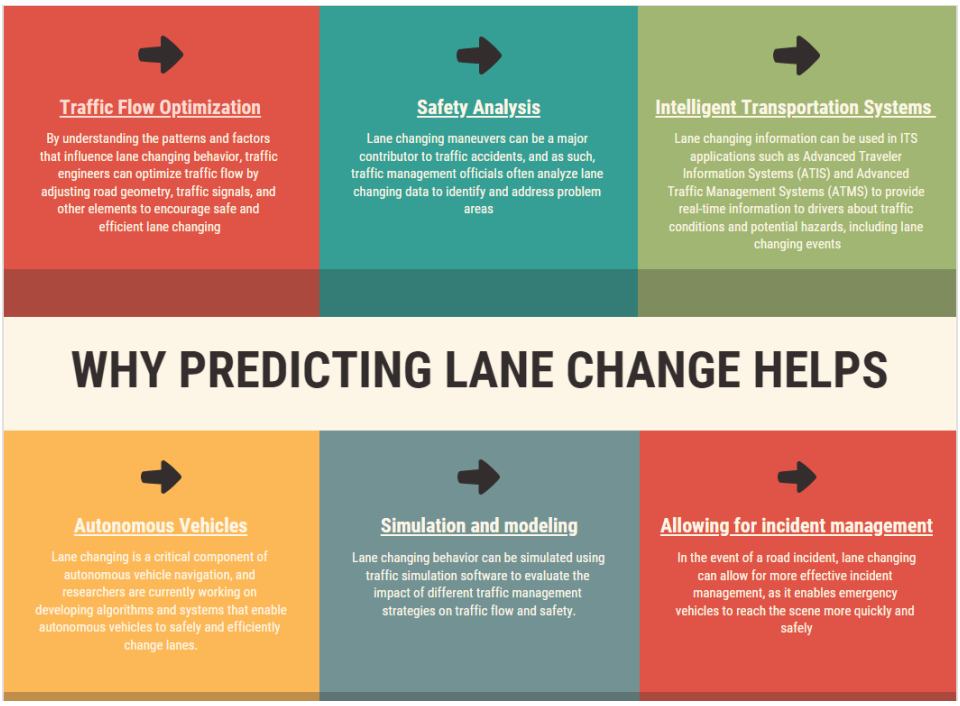


Figure 1. Ways in which lane changing can be used in traffic management.

ITS aims to optimize the usage of existing infrastructure and reduce traffic congestion by improving traffic flow and reducing travel time [12,13]. The applications of lane-changing in traffic management are varied and include both proactive and reactive measures to optimize traffic flow, improve safety, and better serve the needs of drivers and other road-users. However, the task of predicting lane changes is difficult due to the complexity of driving conditions, the unpredictable nature of other drivers, and the scarcity of annotated data. There are many methods that can be used to anticipate the driver's intentions before a changing maneuver. Model-based, trajectory-based, and rule-based change-prediction approaches have all been put forth in the past. Although prior research has improved our ability to anticipate lane-change manoeuvres on highways, there still exists a significant gap in the ability to accurately predict such maneuvers under naturalistic driving scenarios.

1. Rule-based approaches rely on the explicit definition of rules based on the road infrastructure and the traffic situation.
2. Trajectory-based methods rely on analyzing the historical data of the vehicles.
3. Model-based methods rely on mathematical models to predict the lane change.

Unfortunately, these techniques frequently have trouble with real-world driving situations, demand large datasets, and lack robustness. Hence, in this paper, we investigate real-world vehicle trajectories captured on German motorways [14] propose a lane-change prediction model that makes use of Bi-directional LSTM networks. Our proposed model takes advantage of the temporal dependencies between successive observations, capturing the non-linear relationships between various factors that influence lane changes, and can better handle complex driving scenarios. The contributions of the study include:

1. Bidirectional long short-term memory (BiLSTM) network is used to assess naturalistic vehicle trajectories captured from several sensors on German motorways and is offered as a new lane change prediction model.

2. A sliding window approach is introduced to handle the temporal aspect of vehicle behavior, considering both the preceding and following vehicles' trajectories to optimise the predictability of lane changes.
3. The issue of class imbalance in the data is addressed by introducing rolling mean computed weights, resulting in a comprehensive feature set being used to train the model and achieving high accuracy and precision in identifying the two target classes.

The remaining of this article is distributed into the sections that follow: The paper's literature review provides a summary of earlier investigations and studies on the subject. It highlights knowledge gaps and establishes the background for the current research. The research's methodology and procedures, including the dataset preprocessing and model specifics, are described in the study design and methods section. The research's findings are presented in the results section. The discussion section compares our performance with baseline papers. The conclusion section highlights the study's key findings and offers suggestions for more research.

2. Literature Review

Lane-change prediction is a challenging task that has been studied in the discipline of transportation and intelligent transportation systems (ITS) for many years. Numerous methods have been proposed for lane-change prediction. Figure 2 shows the classification of these methods.

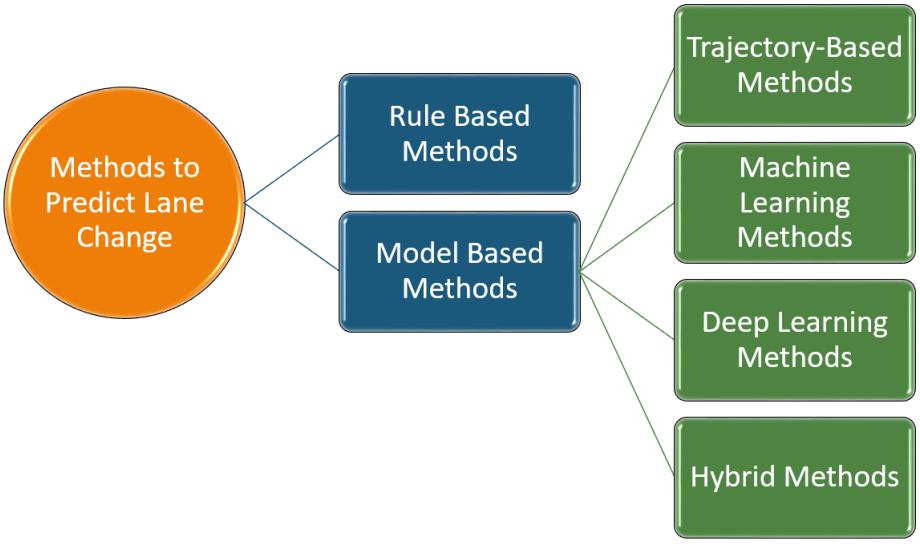


Figure 2. Classification of Methods Used So Far For Predicting Lane Change Maneuver.

2.1. Rule-Based Methods

This prediction method often relies on an explicit definition of rules based on the road infrastructure and the traffic situation. Ref. [15] uses both vertical and horizontal motion cues to predict lane-change movements in driver monitoring systems. The algorithm was tested using NGSIM data on vehicle trajectory and showed high accuracy in recognizing the intentions of left and right lane-change maneuvers with a low false-positive rate. The algorithm is considered suitable for use in ADAS and autonomous vehicles. Previously, Ref. [16] presented a model (MOBIL) for determining lane-changing rules for car-following models. This takes into account the utility of a lane as well as the risk associated with lane changes, as well as safety and incentive criteria, to derive lane-changing rules. The "politeness factor" allows for the consideration of cooperative driving behavior. The model was validated through traffic simulations, and the lane-changing rate was investigated in an open system with an off-ramp.

2.2. Model-Based Methods

Mathematical models are the most used model to predict lane changes. Ref. [17] employed an enhanced Kalman filter to update a road model based on the space continuity of the lane structure. The width of the search area was estimated from the covariances of the model parameters and, with each update, the width becomes narrower, which helps to detect distant lane boundaries more accurately and exclude noisy image features. This results in a more robust lane-recognition system. Ref. [18] is a model-based method to predict lane changes. In this model, the lane-change decision-making process of a driver is modeled using fuzzy logic and inference rules. The model takes into account various factors such as traffic conditions, driving comfort, and driving aggressiveness to make predictions about lane-changing behavior. This approach is considered model-based because it represents the underlying mechanisms and relationships that govern the lane-change decision-making process in a mathematical form. Model-based methods are often used in predictive modeling and control problems to represent and make predictions about a system's behavior. Ref. [19] focuses on developing mandatory lane-changing models for both traditional and connected environments. The study employs the game theory approach and addresses the issues in previous models. The results show that the developed models have high accuracy in replicating observed mandatory lane-changing behavior and outperform existing models. The comparison of the models with Liu's [20] and Talebpour's [21] models indicate that the developed models (AZHW models) have consistently performed better.

2.3. Trajectory Based Methods

Methods based on trajectory analysis look at the vehicles' prior performance information. Ref. [22] The potential field approach is used to generate a trajectory for the target vehicle, such as nearby cars, the goal, and the sidelines of the current lane. The extracted features based on the predicted trajectory are used to re-estimate the driving intention and detect lane changes. To prevent potential crashes during the lane-change process, Ref. [23] suggests a dynamic, automated lane-change manoeuvre. The approach uses an all-encompassing trajectory planning technique that accounts for comfort, safety and traffic efficiency, and transforms the issue into a limited optimization problem based on the time and distance of lane changes. A safe driving distance is maintained by updating a reference trajectory with V2V communication and tracking it with a sliding mode controller. The strategy has been tested and proven effective in simulations and a driving simulator.

2.4. Machine-Learning-Based Methods

Ref. [24] presents a unique Bayesian optimization lane -hange decision-making model for autonomous vehicles that is based on Support Vector Machines (SVM). The model's improved accuracy of 85.33 percent in the training set and 86.27 percent in the test set was used in comparison to a rule-based lane-change model. In terms of true-positive and false-negative accuracy, the SVM model fared better than the rule-based model, proving that it can faithfully describe drivers' decision-making tendencies. The model's accuracy and validity were confirmed in a real-car experiment that served as its validation. Ref. [25] proposed a machine learning model, specifically an adaptive fuzzy neural network, to analyze sensor data such as cameras, LIDAR, and radar to identify patterns and make predictions about lane changes. The authors used a driving simulator to test their method and they showed that their approach can effectively predict lane change events in real-time. They also showed that the adaptive fuzzy neural network is able to improve its performance over time as it is exposed to more data. Ref. [26] proposes a trajectory prediction approach for a lane-changing vehicle in advanced driver assistance systems (ADAS) considering the driver's high-level status. The approach is based on Hidden Markov Models (HMMs) for driving behavior estimation and classification. At the start of the lane-change procedure, the vehicle state emissions are observed in order to estimate the driver status. The next step is to forecast the future trajectory of the lane-changing vehicle using the driver condition

and position. The classifier demonstrated a good performance in identifying the driver's status and was developed and evaluated using real-life driving data. This trajectory prediction method, which can be applied to both self-driving vehicles and early warning systems, generates multiple trajectories based on the classifier's outputs.

2.5. Deep-Learning-Based Methods

Methods based on deep learning are widely used to predict lane changes on highways. A variety of neural network architectures have been proposed and applied in this domain, including fully connected networks, multi-layer perceptrons, and more recently, convolutional neural networks (CNNs) combined with recurrent neural networks (RNNs) or long short-term memory (LSTM) units [27–31]. These models leverage the power of deep learning to process and analyze large amounts of data and make accurate predictions of lane changes. These techniques have shown promising results and have become a key tool in the field of intelligent transportation systems. Ref. [32] suggested a model for capturing driving behavior in lane-change predictions. The model integrates two key components, including a deep belief Nets for decision-making related to lane changes and an lstm for capturing the execution of the lane change. By combining these elements, the model is able to accurately represent driving behavior and make effective predictions of lane changes. The findings demonstrate that the framework accurately predicts LC behaviour, and the sensitivity analysis demonstrates that the preceding vehicle's relative position in the target lane is the most important factor in determining LCD. The authors of [33] introduce a novel deep learning model called direction convolutional LSTM (DCLSTM) that aims to predict a driver's lane-changing behavior. This model takes the lateral trajectory and its spectrum as inputs, and is shown to outperform traditional LSTM methods in accurately predicting the generation and direction of lane-changing intentions. Ref. [27] uses deep learning to predict lane-change maneuvers in a vehicle. The authors use an image dataset, the PREVENTION dataset, to train two different lane-change prediction algorithms: one using a GoogleNet and LSTM model and the other using a trained CNN. The results show that the GoogleNet and LSTM model outperforms the trained CNN, and that using the double-vehicle-size ROI selection method and an extended feature vector slightly increases the classification performance. The authors suggest that future work could include a more multifaceted evaluation of the training and input configuration values, as well as incorporate LiDAR and radar information to improve the results. Ref. [34] presents an improved LSTM approach for lane-change intention prediction for autonomous vehicles, addressing the issue of imbalanced data by using a hierarchical over-sampling bagging method and a sampling technique that keeps the temporal information. Empirical findings using two benchmark datasets demonstrate that in terms of prediction time, the proposed method outperforms numerous baseline algorithms in terms of F1, and G-mean. This method additionally considers the interactions between nearby cars.

2.6. Hybrid Methods

The method proposed in the paper [35] is not a purely rule-based method of lane-change prediction, but rather a combination of a rule-based approach and a deep Q-network (DQN). The proposed algorithm uses rule-based constraints to ensure that the decisions made by the DQN algorithm are safe and can be used in real-world scenarios. The DQN algorithm is trained to take appropriate actions based on sensor data and predefined rules. It should be noted that DQN is a type of reinforcement learning algorithm, which allows the system to learn from experience and adapt to changing traffic conditions. Therefore, it is not a rule-based method, but rather a combination of rule-based and learning-based methods. Some other relevant studies include [36–40]. As can be seen from the above literature, research on lane-change prediction has been extensive in recent years. Therefore, Table 1 provides a concise summary of various approaches and results from recent studies in this area.

Table 1. Predicting lane-changing maneuvers: a snapshot of recent research.

Reference	Year	Approach/Method	Model Architecture	Dataset	Limitations
[41]	2023	Machine Learning	Linear Discriminant, Logistic model, Decision Trees, Naïve Bayes, SVM, ANN,	HighD	The machine learning predictions' configuration, training, and reproducibility still need to be checked and assessed.
[42]	2022	Deep Learning	CNN with spatial attention with Multi-Task Learning approach	German highways dataset	The proposed model is evaluated on only one dataset and the effectiveness of the model in various driving environments and conditions needs to be investigated
[43]	2021	Machine learning	Random Forest (RF)	NGSIM US Route 101 dataset	No limitations mentioned in the paper
[44]	2020	Hybrid	Density-based clustering and SVM for labeling, LSTM network for prediction	HighD	Data collection limited to short highway segment, potential need for validation with other sensors, longer road segments could provide more interactions
[45]	2017	Machine Learning with measurements from on-board sensors.	ANN, SVM	Data collected using a simulation tool	Information from on-board sensors is a challenge for accurate classification. The proposed algorithm may require further validation in real-world driving situations.

3. Study Design and Methods

This section describes the procedures and techniques used in this study to analyze the time series data of highway vehicles. The data were pre-processed to ensure their validity and consistency for analysis. To capture the temporal dynamics of the data, a sliding window approach was employed. This involved dividing the time series data into overlapping windows of 20 time steps, with each window representing the movement of a vehicle. The choice of window size was based on the observation that, on average, 25 frames or rows were present for each vehicle in the data. The sliding window approach allowed for the model to consider the history of each vehicle in its predictions. The data from each window were then fed into a bi-directional long short-term memory (Bi-LSTM) model to predict the lane-change intention of the vehicles. The flow diagram of the overall process is provided in Figure 3 to help understand the steps involved in our study.

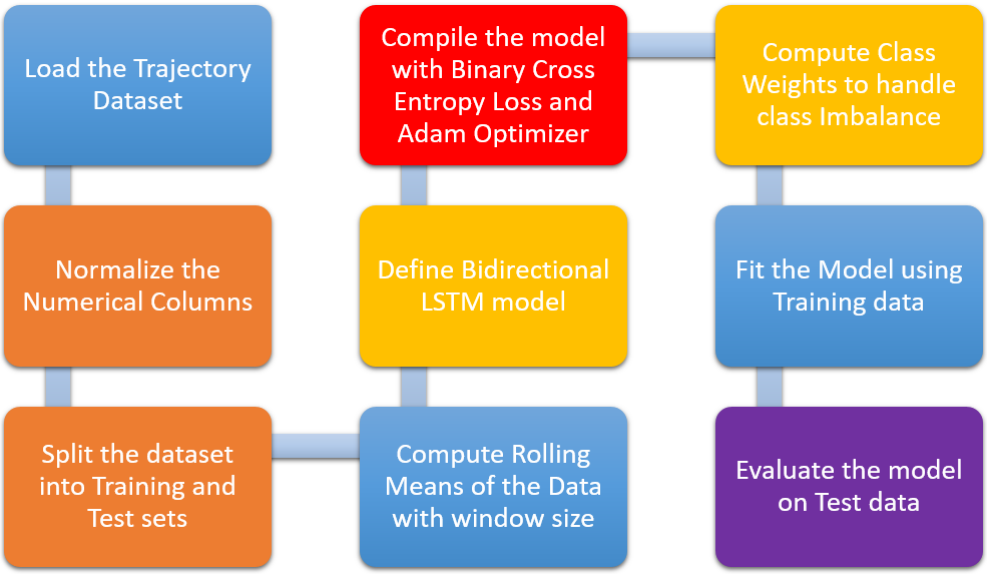


Figure 3. Flow Diagram Illustrating the Overall Process of Lane-Change Prediction in Time Series Data.

3.1. Data Pre-Processing

In data pre-processing, the trajectory dataset used in this study was loaded and merged from 60 different tracks of vehicles on the highway. The original data included vehicle trajectory attributes for each frame, and a target column indicating whether the vehicle changed lanes was added. The data were then normalized using min–max normalization,

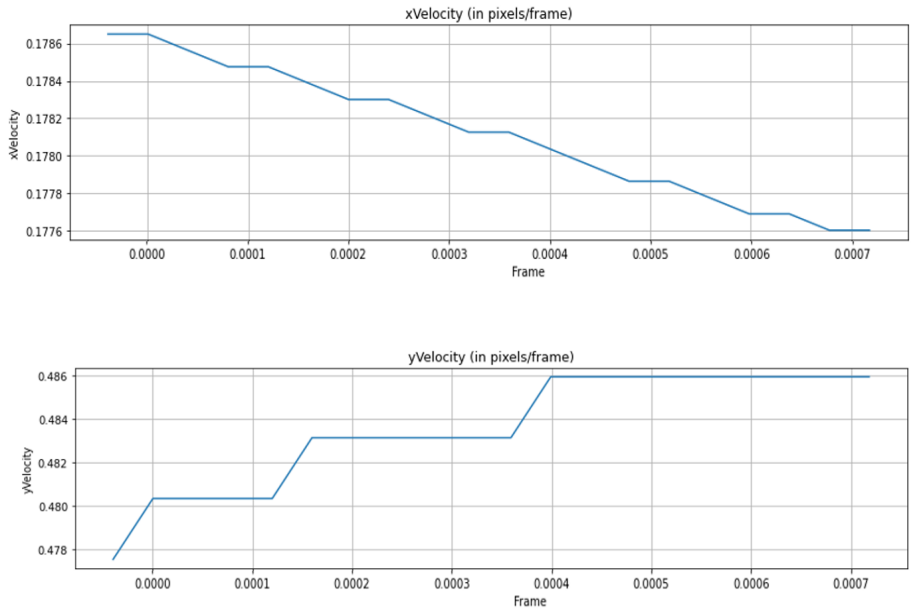
and observations were made regarding the y velocity component for vehicles that changed lanes versus those that did not.

3.1.1. Load the Trajectory Dataset

The dataset used in this study consisted of 60 different tracks of vehicles on the highway [14]. The original data included the vehicle trajectory attributes for each frame, including the vehicle's ID, x, y, x velocity, y velocity, x acceleration, y acceleration. The target column, indicating whether the vehicle changed lanes, was added by observing the vehicles' trajectories. For vehicles that never changed lanes, the target was marked as "No". In case of a single lane-change, the target was marked as "Yes" for all the rows of that vehicle ID. In order to merge all the data from the 60 sensors into one dataset, a sensor column was added.

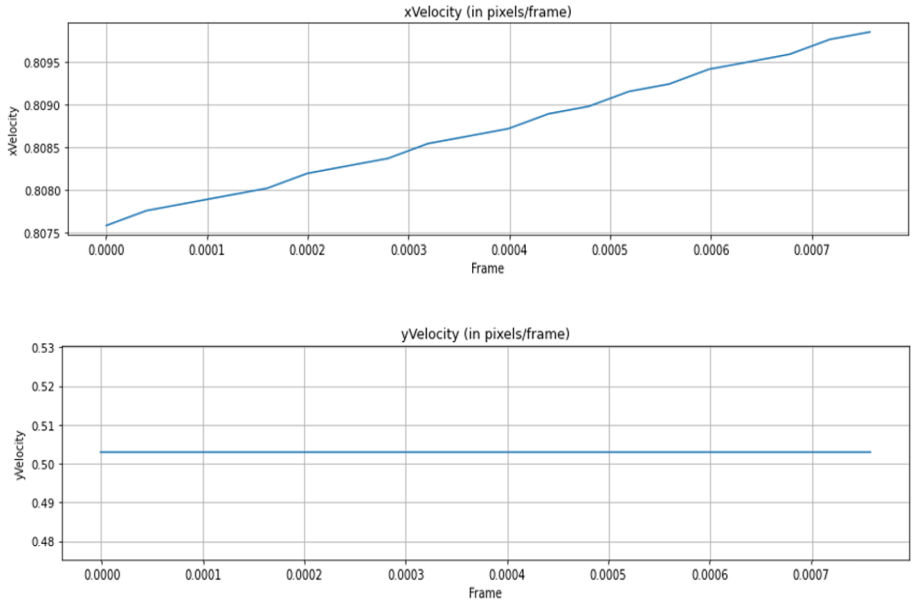
3.1.2. Normalize the Numerical Columns

The data were then normalized using the min–max normalization method to ensure all the attributes were on the same scale. During the data pre-processing, it was observed that the y velocity component of vehicles that did not change lanes was a straight line on the *x*-axis, indicating a lack of variation in *y* velocity. However, for vehicles that did change lanes, the y velocity component was observed as a step-by-step curve, indicating a change in y velocity. These observations were plotted and can be seen in the normalized charts of y and x velocity for target "Yes" and target "No". The study applied the sliding window technique and used Bi-LSTM to predict vehicle lane-change intention. Figure 4a,b shows the comparison of normalized horizontal and vertical velocity components over 20 frames.



(**a**)

Figure 4. *Cont.*



(**b**)

Figure 4. Normalized Trajectory Data Comparison for Vehicles with and without Lane Change. (**a**) with lane change. (**b**) without lane change.

3.1.3. Split the Dataset into Training & Test Sets

The dataset was divided into the training set and the test set for our study. This gave us the opportunity to assess how well our machine learning model performed using fresh, untested data. We divided the dataset using a common technique called random sampling. The dataset was randomly divided between training and test sets with a predetermined ratio using scikit-learn's train_test_split() function. Specifically, we set the test_size parameter to 0.2, which allocated 20% of the data to the test set and the remaining 80% to the training set. We also set the random_state parameter to a fixed value to ensure reproducibility. We were able to train our machine learning model on the training set and assess its performance on the test set by dividing the dataset into training and test sets.

3.2. Sliding Window Approach

In this study, the objective was to analyze the temporal dynamics of the time series data collected from a highway, including information about the vehicle's movement (frame ID, x, y, x velocity, y velocity, x acceleration, y acceleration, and target). To achieve this goal, we employed a sliding window approach and a bidirectional long-short term memory (Bi-LSTM) model.

Compute Rolling Means of the Data with Window Size

The sliding window approach involves dividing the time series data into overlapping windows of 25 time steps. This was chosen as each vehicle had an average of 25 frames of movement information in the data. By using the sliding window approach, each window of data was treated as a separate sample and fed into the Bi-LSTM model to predict the vehicle's lane-change intention. This allowed us to consider the history of each vehicle in the model.

The reason we computed rolling means with a sliding window was to capture the temporal dynamics and patterns present in the time series data. In other words, by analyzing the data in small windows, we could examine how the data changed over time, and identify patterns that were not apparent when looking at the data as a whole.

In the context of the Bi-LSTM model used in this study, computing rolling means that a sliding window allowed for us to capture the historical context of each vehicle's movement behavior. By dividing the time series data into windows of 25 time steps and feeding them

into the model, we were able to provide the model with information on how the vehicle behaved over the previous 25 time steps. This historical context can be critical in predicting the vehicle's future behavior, such as a lane-change intention.

Furthermore, the sliding window approach allowed for us to capture changes in behavior that occur over short time intervals, such as sudden lane changes or other quick maneuvers. By analyzing the data in small windows, we could identify these changes more easily and accurately than by looking at the data as a whole.

3.3. Bi-Directional LSTM

The bidirectional long short-term memory (BiLSTM) model is a type of recurrent neural network designed to analyze sequential data such as time series, speech, or text. In this BiLSTM model, two separate LSTMs were trained, one in the forward direction and another in the backward direction, to capture contextual information in both directions. The outputs from both the LSTMs were then concatenated to produce the final output. Figure 5 shows the general architecture of a BiLstm network.

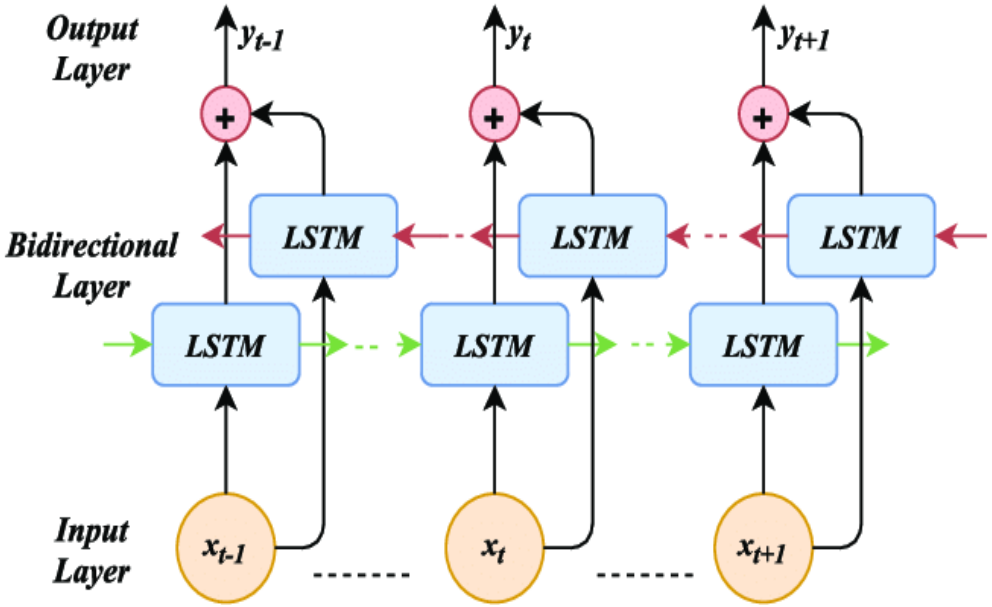


Figure 5. Architecture of BiLSTM Network adapted from [46].

For the task of lane-change prediction, the BiLSTM model can be used to process the sequential data collected from the vehicle's sensors, such as GPS, speed, acceleration, and yaw rate. This sequential data can then be used to learn the patterns and dependencies between different variables, allowing the model to make predictions about the likelihood of a lane change occurring. One of the main reasons why BiLSTM was chosen for this task is its ability to handle sequences of varying lengths and its ability to capture both past and future contextual information. This is important in the context of lane-change prediction as it allows the model to consider both the previous driving behavior of the vehicle and any future events that may impact the vehicle's trajectory. Another reason is the bidirectional architecture of the BiLSTM, which enables the model to take consider the direction of the sequential data. This can be useful in the context of lane-change prediction, as the direction of the vehicle's motion can have a significant impact on its likelihood of changing lanes. Overall, the BiLSTM model provides a flexible and powerful approach to analyze sequential data and make predictions based on these data, making it well-suited to the task of lane-change prediction.

3.3.1. Define BiDirectional LSTM Model

In this section, we present the architecture of our proposed BiDirectional LSTM model, as depicted in Figure 6. In addition to the architecture of the proposed BiDirectional LSTM model shown in Figure 6, we also provide the corresponding algorithm in Algorithm 1 to demonstrate the specific steps involved in training the model. Below, we introduce and define each of the model's architecture layer in detail.

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 bidirectional (Bidirectiona  (None, 9, 256)           133120
 l)

 batch_normalization (BatchN  (None, 9, 256)           1024
 ormalization)

 dropout (Dropout)           (None, 9, 256)            0

 bidirectional_1 (Bidirectio  (None, 128)              164352
 nal)

 batch_normalization_1 (Batc  (None, 128)              512
 hNormalization)

 dropout_1 (Dropout)         (None, 128)               0

 dense (Dense)               (None, 64)                8256

 dropout_2 (Dropout)         (None, 64)                0

 dense_1 (Dense)             (None, 32)                2080

 dropout_3 (Dropout)         (None, 32)                0

 dense_2 (Dense)             (None, 1)                 33

=================================================================
Total params: 309,377
Trainable params: 308,609
Non-trainable params: 768
_________________________________________________________________
```

Figure 6. Summary of the Bidirectional LSTM Model used for Lane-Change Prediction. The model consists of multiple layers, including bidirectional LSTMs.

Bidirectional LSTM Layer

This layer consisted of 128 hidden units and was configured to return sequences (i.e., output at every time step) using the "return_sequences" parameter set to True. The layer also used L2 regularization with a strength of 0.001 to prevent overfitting.

Batch Normalization Layer

By dividing by the standard deviation and removing the mean, this layer normalised the output of the preceding layer. This enhanced the model's performance and helped to stabilise the training process.To avoid overfitting, the output of this layer was also subjected to a dropout layer with a dropout rate of 0.1.

Second Bidirectional LSTM Layer

This layer consisted of 64 hidden units and used L2 regularization with a strength of 0.1. The output of this layer was fed into another batch normalization layer, followed by a dropout layer with a dropout rate of 0.1.

Dense Layers

The model then had three dense (fully connected) layers with 64, 32, and 1 neurons, respectively. The first two dense layers used the ReLU activation function, while the final layer used the sigmoid activation function. The dropout rate for each dense layer was set to 0.1 to prevent overfitting.

We defined "hidden units" as the number of neurons in a layer that were not directly connected to the input or output of the layer. These neurons performed computations on

the input data and produced an output that was passed on to the next layer. Overall, this architecture aims to optimize the performance of the model while avoiding overfitting.

Algorithm 1 Bidirectional LSTM Model

Require: Training data $\mathbf{X}_{train}$, Training labels $\mathbf{y}_{train}$, Test data $\mathbf{X}_{test}$, Test labels $\mathbf{y}_{test}$, Learning rate α, Regularization parameter λ
Ensure: Trained Bidirectional LSTM model
1: Compute rolling mean for the training data
2: Initialize window size w to 25
3: Initialize weights w_i to $\frac{1.0}{w}$ for all $i = 1, \ldots, w$
4: Apply rolling window with size w to training data to compute rolling mean
5: Create new training data dataframe df using rolling mean
6: Initialize activity regularization parameter to λ
7: Create new sequential model model
8: Add Bidirectional LSTM layer with 128 units and input shape of $\mathbf{X}_{train}$ to model
9: Add Batch Normalization layer to model
10: Add Dropout layer with dropout rate of 0.1 to model
11: Add Bidirectional LSTM layer with 64 units to model
12: Add Batch Normalization layer to model
13: Add Dropout layer with dropout rate of 0.1 to model
14: Add Dense layer with 64 units and ReLU activation to model
15: Add Dropout layer with dropout rate of 0.1 to model
16: Add Dense layer with 32 units and ReLU activation to model
17: Add Dropout layer with dropout rate of 0.1 to model
18: Add Dense layer with 1 unit and Sigmoid activation to model
19: Print summary of model
20: Define root mean squared error function using Keras backend
21: Compile model with binary crossentropy loss, Adam optimizer, accuracy metric, and root mean squared error metric
22: Convert training and test labels from strings to integers using LabelEncoder
23: Compute class weights using balanced class weighting method
24: Create dictionary of class weights with class indices as keys and class weights as values
25: Initialize TensorBoard callback for model training logs
26: Fit model on training data and labels with 20 epochs, batch size of 90, validation data of test data and labels, class weights dictionary, and TensorBoard callback

3.3.2. Compile the Model

In our research, we compiled a neural network model by configuring the optimizer, loss function, and evaluation metrics. The choice of optimizer and loss function was dependent on the type of problem being solved, while the evaluation metrics were used to assess the performance of the model during training and testing. For our specific problem of binary classification, we used the binary cross-entropy loss function, which measures the difference between the predicted probability and the true label (either 0 or 1). We selected the Adam optimizer, which is a popular choice in deep learning because it combines the benefits of two other optimization methods: AdaGrad and RMSProp. To assess the performance of our model during training and testing, we used the accuracy metric. This metric measures the percentage of correctly classified samples. Additionally, we also used the root mean squared error (RMSE) metric to evaluate our model. Although RMSE is typically used for regression problems, we included it as an additional evaluation metric to gain a more comprehensive understanding of our model's performance.In order to assure the best performance for our particular challenge of binary classification, we carefully considered the optimizer, loss function, and evaluation metrics when building our neural network model.

3.3.3. Compute Class Weights to Deal with Class Imbalance

In our study, we encountered the problem of detecting lane-change maneuvers using a trajectory dataset. Since lane-change maneuvers occur relatively infrequently compared to other driving maneuvers, the resulting dataset was highly imbalanced, with a majority of samples representing non-lane-change scenarios. To address this issue, we computed class weights for the dataset using the compute_class_weight() function from the scikit-learn library. This function considers the class distribution of the dataset and computes weights that assign greater importance to the minority class during training. In our case, the minority class represented lane-change scenarios, which were of particular interest for our application. By using class weights during training, we were able to reduce the bias towards the majority class and improve the model's ability to accurately classify lane-change scenarios. This is important for safety-critical applications such as lane-departure warning systems, where detecting lane changes accurately can help prevent accidents and improve overall road safety. In conclusion, generating class weights enabled us to address the class imbalance problem in our trajectory dataset and enhance the performance of our model in identifying lane-change manoeuvres.

3.3.4. Fit the Model Using Training Data

Once we compiled our neural network model, we fit it with the training data using the fit() function from the Keras API. The fit() function trains the model by iterating over the training data for a specified number of epochs, updating the model's weights at each iteration to minimize the loss function. In our study, we used the fit() function to train the model for 1000 epochs, with a batch size of 90. We also included a validation dataset consisting of the X_test and y_test data to monitor the model's performance during training. To deal with the issue of class imbalance in our dataset, we used the class_weight parameter in the fit() function to assign greater importance to the minority class during training. Specifically, we passed in a dictionary of class weights computed using the compute_class_weight() function from the scikit-learn library. The dict(enumerate(class_weights)) expression maps the computed class weights to the corresponding class indices in the dataset. During training, we also used a tensorboard callback to monitor the model's performance and visualize various metrics, such as loss and accuracy. This helped us to identify potential issues with the model and make adjustments as needed. Overall, iterating over the dataset for a predetermined number of epochs and changing the model's weights to minimise the loss function were usd to fit the model with the training data. We were able to increase the model's accuracy and effectiveness for our particular problem of recognising lane-change movements using a trajectory dataset by employing class weights, monitoring the model's performance with a validation dataset, and using tensorboard and a tensorboard validation dataset.

3.3.5. Evaluate the Model on Test Data

We used the following evaluation metrics to evaluate our model's performance:

Accuracy

The accuracy of a machine learning model is a frequently used statistic to gauge its performance. As a proportion of all predictions, it determines the model's accuracy rate. The number of accurate forecasts and the total number of guesses made make up the numerator and denominator of the fraction or percentage that represents it. For calculating accuracy, use the formula:

$$Accuracy = (Number of Correct Predictions)/(Total Number of Predictions) \quad (1)$$

Precision

Precision gauges the proportion of correct positive predictions among the total number of positive predictions made by the model. A high precision value indicates a low rate of false-positive predictions made by the model.

$$Pr = TruePositives/(TruePositives + FalsePositives) \tag{2}$$

where, Pr = Precision

True positives (TP) are the number of positive cases that are correctly identified as positive.

False negatives (FN) are the number of positive cases that are incorrectly identified as negative.

Recall

Recall counts the number of predictions that came true out of all the actual positive events. A high recall value means that the model is able to identify most of the positive instances, i.e., the model has a low false-negative rate.

$$Rc = TruePositives/(TruePositives + FalseNegatives) \tag{3}$$

where, Rc = Recall

$F1$ Score

The $F1$ score represents the balance between precision and recall and is computed as the harmonic mean of the two metrics. A high score indicates that the model has a good balance between precision and recall, whereas a low value suggests a poor balance.

$$F1 = 2 \times (Pr \times Rc)/(Pr + Rc) \tag{4}$$

AUC

A binary classification model's performance is assessed using this metric. Area under the curve (AUC) measures how easily positive and negative events can be distinguished. AuC with 1.0 denotes the complete separation of positive and negative cases and 0.5 denotes an imperfect separation of positive and negative examples. The formula for calculating is given by:

$$AuC = \frac{1}{n}\sum_{i=1}^{n-1}\frac{(y_i + y_{i+1})}{2}(x_{i+1} - x_i) \tag{5}$$

where n is the number of data points in the ROC curve, (x_i, y_i) are the coordinates of the $i - th$ data point in the ROC space. This curve is a plot of the sensitivity versus the fall-out for a binary classification problem. By calculating the area under the ROC curve, the AUC assesses a classifier's overall performance. The range of AUC values is from 0 to 1, with 1 denoting perfect performance and 0.5 denoting random performance.

Root Mean Square Error (RMSE)

The square root of the mean of the squared discrepancies between the predicted and actual values was used to calculate RMSE. This is a measurement of how far the model's predictions differ from the actual data. The model fits the data better with a lower RMSE. This is due to the fact that a lower RMSE signifies a smaller discrepancy between the predicted and actual values, demonstrating a higher level of model accuracy.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(o_i - p_i)^2} \tag{6}$$

where:

n is sample size/observations
o_i is the actual value
p_i is the predicted value

4. Results

When using the proposed model's training results, two different training methods on the datasets are displayed in Figure 7a–d.

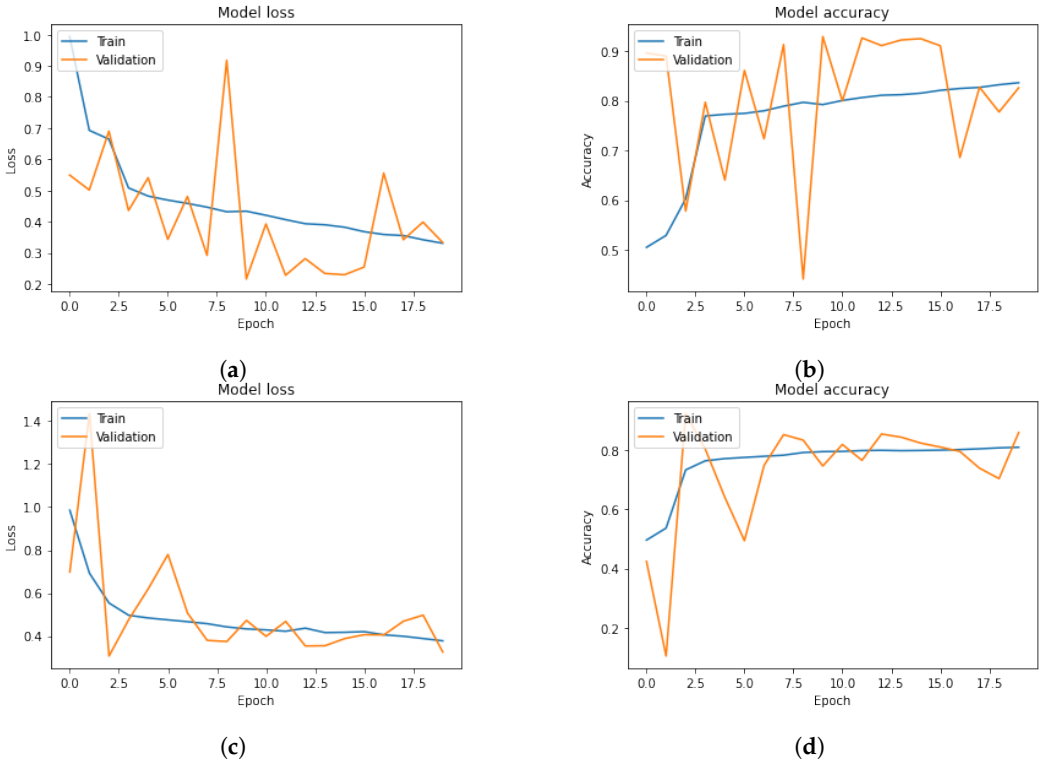


Figure 7. Loss and accuracy curves of two training sessions. (**a**) Loss curve of first training session. (**b**) Accuracy curve of first training session. (**c**) Loss curve of second training session. (**d**) Accuracy curve of second training session.

Table 2 shows the loss and accuracy of two trainings, on the test set and training set, as well as the validation set. Table 3 provides a performance comparison of two training models on a sensor data set. The data set was divided into two halves, and each half was used to train a different model. The table shows the results in terms of accuracy, F1 score, precision, recall, and AUC. The accuracy and F1 score of the first training model performed better, but the recall and precision of the second training model performed better.

Table 2. Comparisonof Loss and Accuracy Metrics.

Metric		Test	Training	Validation
First Training	Loss	0.32	0.41	0.47
	Accuracy	0.86	0.81	0.77
Second Training	Loss	0.33	0.03	0.14
	Accuracy	0.83	0.99	0.96

Table 3. Performance Comparison of Two Training Methods on Test Sets.

	Test Accuracy	F1	Precision	Recall	AUC
First Training	0.83	0.51	0.36	0.89	0.85
Second Training	0.86	0.53	0.41	0.75	0.81

The model was trained on a dataset with 20 epochs and the results were evaluated using the accuracy and loss metrics. The accuracy measures the model's ability to correctly predict lane changes while the loss measures the error between the predicted and actual values. With our first training on half of the sensor data, the training accuracy improved from 49.64% in the first epoch to 81.06% in the final epoch. The training loss also decreased from 0.9850 in the first epoch to 0.3776 in the final epoch. The validation accuracy started at 42.39% in the first epoch and improved to 86.05% in the final epoch. The validation loss started at 0.6981 in the first epoch and decreased to 0.3255 in the final epoch. Here, our model acieved a test accuracy of 86% with a test loss of 0.325. These results suggest that our model was able to learn from the training data and make accurate predictions on the validation data. Also, during training, we noticed a decrease in the root mean square (RMS) value from 7.054 to 2.33, which indicated that our model improved over time. This suggests that our model became better at predicting the target values, and the error between the predicted and actual values gradually decreased. Hence, due to the use of the rolling mean approach and compute weights to address the class imbalance, we helped our model to better handle the imbalanced data by weighing the minority class higher. This resulted in a more accurate model and improved performance metrics with an F1 score of 0.52, a precision of 0.41, a recall of 0.75, an accuracy of 0.86, and an AUC of 0.81 demonstrating how well our model can tell the difference between the lane-changing and keeping classes. On the other half of the data samples, we conducted training and observed that the model achieved an accuracy of 83.65% with a loss value of 0.3306. The validation accuracy was also observed to improve over the epochs, with the highest validation accuracy of 92.53%. The RMSE value decreased from 5.67 to 2.67, indicating the model's improvement over time. The test loss was 0.33 with a test accuracy of 83%. Our model's strong ability to identify positive and negative classes is demonstrated by its F1 score of 0.51 and its precision of 0.36, recall of 0.89, accuracy of 0.82, and AUC of 0.85.

5. Discussion

In order to determine the effectiveness of our suggested methodology, we compare the performance of our lane-change prediction model based on trajectory data with other, comparable baseline methods in this section. The four baseline papers we chose for this purpose are as follows:

1. Ref. [47]: This work applies the LSTM approach to forecast future longitudinal and lateral trajectories for automobiles on highways, trained on the NGSIM dataset, using local lateral position, local longitudinal position, and local longitudinal position as major features.
2. Ref. [48]: This work combines deep learning based, two-dimensional trajectory prediction models to forecast both car-following and lane-changing behaviours simultaneously. The proposed model to simulate and predict joint behaviours incorporates BiLSTM), a switch neural network structure based on the attention mechanism, and a temporal convolution neural network (TCN). This model was trained and evaluated using the NGSIM dataset.
3. Ref. [49]: This paper utilizes a simple BiLSTM model for path prediction.
4. Ref. [50]: This paper employs a CNN-LSTM hybrid sequential model for the prediction of motion trajectory.

It is worth noting that although these papers use the NGSIM dataset, the characteristics of the data and the objectives of the research are different; hence, the performance cannot be directly compared. Nonetheless, we compared the RMSE value of our method with these baseline methods, and the results are presented in the table. As can be seen in Table 4, our proposed method outperformed all the baseline methods, with the lowest RMSE value.

Table 4. Comparison with Baseline Papers.

Model	Method	RMSE
Our Model	BiLSTM with weighted averages	2.67
[47]	LSTM	9.37
[48]	ITPM	5.67
[49]	BiLSTM	8.36
[50]	CNNLSTM	6.66

6. Conclusions

In conclusion, this study proposed a novel approach for predicting lane-change behavior on highways using Bi-LSTM networks. The preprocessed dataset was modified to fit the lane-change prediction task, and the results demonstrated that the proposed approach achieved high accuracy by considering both the past and future context of the input data. The model achieved an accuracy of 86% on one half of the dataset and 83.65% on the other half, with an F1 score of 0.52 and 0.51, respectively. The precision, recall, accuracy, and AUC also showed that the model had a high discrimination ability between the two target classes. The proposed approach outperformed other models in terms of execution time and simplicity, making it a viable solution for real-time lane-change prediction in practical applications. This research has one limitation that the dataset used is limited to a specific region and time frame, which may affect the generalizability of the model to other regions and time periods. Nevertheless, the proposed approach shows promising results and has the potential to significantly improve highway safety through better vehicle coordination and proactive warnings to drivers. However, there is still room for improvement, and future work will focus on the following:

1. Increasing the dataset to encompass various driving scenarios and settings to enhance the model's generalization capability.
2. Investigating other deep learning architectures to improve the model's performance.
3. Conducting k-fold cross-validation to evaluate the model's performance on multiple subsets of the dataset.

In summary, this study presents a promising approach for lane-change prediction on highways using Bi-LSTM networks. The results demonstrate a high accuracy and discrimination ability, and future work will focus on enhancing the model's performance and evaluating its applicability in real-world situations.

Author Contributions: All authors have equal contributions. All authors have read and agreed to the published version of the manuscript.

Funding: The authors extend their appreciation to the Deputyship for Research & Innovation, Ministry of Education in Saudi Arabia for funding this research work through the project number RI-44-0831.

Data Availability Statement: Data will be available on request.

Acknowledgments: The authors extend their appreciation to the Deputyship for Research & Innovation, Ministry of Education in Saudi Arabia for funding this research work through the project number RI-44-0831.

Conflicts of Interest: Authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ADAS	Advanced Driver-Assistance Systems
AUC	Area Under the Curve
API	Application Programming Interface
ANN	Artificial Neural Network
BILSTM	Bidirectional Long Short-Term Memory
CV	Connected Vehicles
DL	Deep Learning
DQN	Deep Q-Network
FC	Fully Connected
F1	F1 Score
G-Mean	Geometric Mean
GPS	Global Positioning System
HMM	Hidden Markov Models
LCD	Lane Change Decisions
LSTM	Long Short-Term Memory
L2	L2 Regularization
LiDAR	Light Detection and Ranging
RADAR	Radio Detection and Ranging
ReLU	Rectified Linear Unit
RMS	Root Mean Square
RMSE	Root Mean Square Error
ROC	Receiver Operating Characteristic
SVM	Support Vector Machine
ITS	Intelligent Transport System

References

1. Sen, B.; Smith, J.; Najm, W. *Analysis of Lane Change Crashes*; National Highway Traffic Safety Administration: Washington, DC, USA, 2003.
2. Khan, M.; Lee, S. A comprehensive survey of driving monitoring and assistance systems. *Sensors* **2019**, *19*, 2574. [CrossRef] [PubMed]
3. Greenwood, P.; Lenneman, J.; Baldwin, C. Advanced driver assistance systems (ADAS): Demographics, preferred sources of information, and accuracy of ADAS knowledge. *Transp. Res. Part F Traffic Psychol. Behav.* **2022**, *86*, 131–150. [CrossRef]
4. Fagnant, D.; Kockelman, K. Preparing a nation for autonomous vehicles: Opportunities, barriers and policy recommendations. *Transp. Res. Part A Policy Pract.* **2015**, *77*, 167–181. [CrossRef]
5. Haboucha, C.; Ishaq, R.; Shiftan, Y. User preferences regarding autonomous vehicles. *Transp. Res. Part C Emerg. Technol.* **2017**, *78*, 37–49. [CrossRef]
6. Narayanan, S.; Chaniotakis, E.; Antoniou, C. Shared autonomous vehicle services: A compre-hensive review. *Transp. Res. Part C Emerg. Technol.* **2020**, *111*, 255–293. [CrossRef]
7. Hussain, R.; Zeadally, S. Autonomous cars: Research results, issues, and future challenges. *IEEE Commun. Surv. Tutor.* **2018**, *21*, 1275–1313. [CrossRef]
8. Kaffash, S.; Nguyen, A.; Zhu, J. Big data algorithms and applications in intelligent transportation sys-tem: A review and bibliometric analysis. *Int. J. Prod. Econ.* **2021**, *231*, 107868. [CrossRef]
9. Arena, F.; Pau, G.; Severino, A. A review on IEEE 802.11 p for intelligent transportation systems. *J. Sens. Actuator Netw.* **2020**, *9*, 22. [CrossRef]
10. Veres, M.; Moussa, M. Deep learning for intelligent transportation systems: A survey of emerging trends. *IEEE Trans. Intell. Transp. Syst.* **2019**, *21*, 3152–3168. [CrossRef]
11. Saharan, S.; Bawa, S.; Kumar, N. Dynamic pricing techniques for Intelligent Transportation Sys-tem in smart cities: A systematic review. *Comput. Commun.* **2020**, *150*, 603–625. [CrossRef]
12. Humayun, M.; Ashfaq, F.; Jhanjhi, N.; Alsadun, M. Traffic management: Multiscale vehicle detec-tion in varying weather conditions using yolov4 and spatial pyramid pooling network. *Electronics* **2022**, *11*, 2748. [CrossRef]
13. Humayun, M.; Afsar, S.; Almufareh, M.F.; Jhanjhi, N.Z.; AlSuwailem, M. Smart Traffic Management System for Metropolitan Cities of Kingdom Using Cutting Edge Technologies. *J. Adv. Transp.* **2022**, *2022*, 4687319. [CrossRef]

14. Krajewski, R.; Bock, J.; Kloeker, L.; Eckstein, L. The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 2118–2125.
15. Nilsson, J.; Fredriksson, J.; Coelingh, E. Rule-based highway maneuver intention recognition. In Proceedings of the 2015 IEEE 18th International Conference on Intelligent Transportation Systems, Gran Canaria, Spain, 15–18 September 2015; pp. 950–955.
16. Kesting, A.; Treiber, M.; Helbing, D. General lane-changing model MOBIL for car-following models. *Transp. Res. Record* **2007**, *1999*, 86–94. [CrossRef]
17. Takahashi, A.; Ninomiya, Y. Model-based lane recognition. In Proceedings of the Conference on Intelligent Vehicles, Tokyo, Japan, 19–20 September 1996; pp. 201–206.
18. Guan, M.; Wang, Z.; Yang, B.; Nakano, K. A Fuzzy Inference-Based Driver's Lane-Change Decision-Making Model with Different Levels of Aggressiveness. *Proc. Jpn. Jt. Autom. Control Conf.* **2022**, *65*, 604–609.
19. Laval, J.; Daganzo, C. Lane-changing in traffic streams. *Transp. Res. Part B Methodol.* **2006**, *40*, 251–264. [CrossRef]
20. Ban, J. A game theoretical approach for modelling merging and yielding behaviour at freeway on-ramp sec-tions. In *Transportation and Traffic Theory: Papers Selected for Presentation at 17th International Symposium on Transportation and Traffic Theory, a Peer Reviewed Series Since 1959*; Elsevier: Amsterdam, The Netherlands, 2007; p. 197.
21. Talebpour, A.; Mahmassani, H.; Hamdar, S. Modeling lane-changing behavior in a connected environment: A game theory approach. *Transp. Res. Procedia* **2015**, *7*, 420–440. [CrossRef]
22. Woo, H.; Ji, Y.; Kono, H.; Tamura, Y.; Kuroda, Y.; Sugano, T.; Yamamoto, Y.; Yamashita, A.; Asama, H. Lane-change detection based on vehicle-trajectory prediction. *IEEE Robot. Autom. Lett.* **2017**, *2*, 1109–1116. [CrossRef]
23. Luo, Y.; Xiang, Y.; Cao, K.; Li, K. A dynamic automated lane change maneuver based on vehi-cle-to-vehicle communication. *Transp. Res. Part C Emerg. Technol.* **2016**, *62*, 87–102. [CrossRef]
24. Liu, Y.; Wang, X.; Li, L.; Cheng, S.; Chen, Z. A novel lane change decision-making model of autonomous vehicle based on support vector machine. *IEEE Access* **2019**, *7*, 26543–26550. [CrossRef]
25. Tang, J.; Liu, F.; Zhang, W.; Ke, R.; Zou, Y. Lane-changes prediction based on adaptive fuzzy neural net-work. *Expert Syst. Appl.* **2018**, *91*, 452–463. [CrossRef]
26. Liu, P.; Kurt, A.; Özgüner, Ü. Trajectory prediction of a lane changing vehicle based on driver behav-ior estimation and classification. In Proceedings of the 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), Macau, China, 8–12 October 2022; pp. 942–947.
27. Izquierdo, R.; Quintanar, A.; Parra, I.; Fernández-Llorca, D.; Sotelo, M. Experimental vali-dation of lane-change intention prediction methodologies based on CNN and LSTM. In Proceedings of the 2019 IEEE Intelligent Transportation Systems Conference (ITSC), Auckland, New Zealand, 27–30 October 2019; pp. 3657–3662.
28. Saranya, M.; Archana, N.; Reshma, J.; Sangeetha, S.; Varalakshmi, M. Object detection and lane changing for self driving car using cnn. In Proceedings of the 2022 International Conference on Communication, Computing and Internet of Things (IC3IoT), Chennai, India, 10–11 March 2022; pp. 1–7.
29. Mo, X.; Xing, Y.; Lv, C. Interaction-aware trajectory prediction of connected vehicles using cnn-lstm networks. In Proceedings of the IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society, Singapore, 18–21 October 2020; pp. 5057–5062.
30. Zhang, Y.; Zhang, S.; Luo, R. Lane Change Intent Prediction Based on Multi-Channel CNN Consider-ing Vehicle Time-Series Trajectory. In Proceedings of the 17th International IEEE Conference On Intelligent Transportation Systems (ITSC), Macau, China, 8–12 October 2022; pp. 287–292.
31. Lee, D.; Kwon, Y.; McMains, S.; Hedrick, J. Convolution neural network-based lane change inten-tion prediction of surrounding vehicles for ACC. In Proceedings of the 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC), Yokohama, Japan, 16–19 October 2017; pp. 1–6.
32. Xie, D.; Fang, Z.; Jia, B.; He, Z. A data-driven lane-changing model based on deep learning. *Transp. Res. Part C Emerg. Technol.* **2019**, *106*, 41–60. [CrossRef]
33. Zhao, N.; Wang, B.; Lu, Y.; Su, R. Direction convolutional LSTM network: Prediction network for drivers' lane-changing behaviours. In Proceedings of the 2022 IEEE 17th International Conference on Control Automation (ICCA), Naples, Italy, 27–30 June 2022; pp. 752–757.
34. Shi, Q.; Zhang, H. An improved learning-based LSTM approach for lane change intention prediction subject to imbalanced data. *Transp. Res. Part C Emerg. Technol.* **2021**, *133*, 103414. [CrossRef]
35. Wang, J.; Zhang, Q.; Zhao, D.; Chen, Y. Lane change decision-making through deep reinforcement learning with rule-based constraints. In Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, 14–19 July 2019; pp. 1–6.
36. Torkey, H.; Atlam, M.; El-Fishawy, N.; Salem, H. Machine Learning Model for Cancer Diagnosis based on RNAseq Microarray. *Menoufia J. Electron. Eng. Res.* **2021**, *30*, 65–75. [CrossRef]
37. Torkey, H.; Atlam, M.; El-Fishawy, N.; Salem, H. A novel deep autoencoder based survival analysis approach for microarray dataset. *PeerJ Comput. Sci.* **2021**, *7*, e492. [CrossRef]
38. Zhao, L.; Xu, T.; Zhang, Z.; Hao, Y. Lane-Changing Recognition of Urban Expressway Exit Using Natural Driving Data. *Appl. Sci.* **2022**, *12*, 9762. [CrossRef]

39. Zhang, H.; Guo, Y.; Wang, C.; Fu, R. Stacking-based ensemble learning method for the recognition of the preceding vehicle lane-changing manoeuvre: A naturalistic driving study on the highway. *IET Intell. Transp. Syst.* **2022**, *16*, 489–503. [CrossRef]
40. Liu, J.; Liu, Y.; Wei, D.; Ni, W.; Zeng, X.; Song, L. Attention-Based Auto-Encoder Framework for Ab-normal Driving Detection. In Proceedings of the 2022 IEEE International Symposium on Circuits and Systems (ISCAS), Austin, TX, USA, 27 May 2022–1 June 2022; pp. 3150–3154.
41. Khelfa, B.; Ba, I.; Tordeux, A. Predicting highway lane-changing maneuvers: A benchmark analy-sis of machine and ensemble learning algorithms. *Phys. A Stat. Mech. Its Appl.* **2023**, *612*, 128471. [CrossRef]
42. Mozaffari, S.; Arnold, E.; Dianati, M.; Fallah, S. Early lane change prediction for automated driving systems using multi-task attention-based convolutional neural networks. *IEEE Trans. Intell. Veh.* **2022**, *7*, 758–770. [CrossRef]
43. Abraham, A.; Zhang, Y.; Prasad, S. Real-time prediction of multi-class lane-changing intentions based on highway vehicle trajectories. In Proceedings of the 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), Indianapolis, IN, USA, 19–22 September 2021; pp. 1457–1462.
44. Mahajan, V.; Katrakazas, C.; Antoniou, C. Prediction of lane-changing maneuvers with auto-matic labeling and deep learning. *Transp. Res. Record* **2020**, *2674*, 336–347. [CrossRef]
45. Kim, I.; Bong, J.; Park, J.; Park, S. Prediction of driver's intention of lane change by augmenting sensor information using machine learning techniques. *Sensors* **2017**, *17*, 1350. [CrossRef]
46. Ihianle, I.; Nwajana, A.; Ebenuwa, S.; Otuka, R.; Owa, K.; Orisatoki, M. A deep learning approach for human activities recognition from multimodal sensing devices. *IEEE Access* **2020**, *8*, 179028–179038. [CrossRef]
47. Altché, F.; La Fortelle, A. An LSTM network for highway trajectory prediction. In Proceedings of the 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC), Yokohama, Japan, 16–19 October 2017; pp. 353–359.
48. Shi, K.; Wu, Y.; Shi, H.; Zhou, Y.; Ran, B. An integrated car-following and lane changing vehicle tra-jectory prediction algorithm based on a deep neural network. *Phys. A Stat. Mech. Its Appl.* **2022**, *599*, 127303. [CrossRef]
49. Abdalla, M.; Hendawi, A.; Mokhtar, H.; Elgamal, N.; Krumm, J.; Ali, M. DeepMotions: A Deep Learning System for Path Prediction Using Similar Motions. *IEEE Access* **2020**, *8*, 23881–23894. [CrossRef]
50. Xie, G.; Shangguan, A.; Fei, R.; Ji, W.; Ma, W.; Hei, X. Motion trajectory prediction based on a CNN-LSTM sequential model. *Sci. China Inf. Sci.* **2020**, *63*, 1–21. [CrossRef]