applied sciences

MDPI

Article

Developing Artificial Intelligence-Based Car-Following Models Using Improved Permutation Entropy Analysis Results

Ali Muhssin Shahatha * and İsmail Şahin

Department of Civil Engineering, Yildiz Technical University, Istanbul 34220, Türkiye; sahin@yildiz.edu.tr
* Correspondence: alimuhssin83@gmail.com

Abstract

Vehicle trajectories are time series, and entropy is a powerful tool for testing or quantifying the complexity of a given series. Entropy tools are often applied to variables such as velocity, acceleration, space headway, and time headway, but the local position data have not been addressed previously. The novelty of this study is that it uses the Improved Permutation Entropy (IPE) for the first time to analyze vehicle position data and convert those data into a limited range (0–0.3317), aiming to understand individual vehicle behavior during car-following and introduce a new prediction method for developing artificial intelligence-based car-following models. The study uses the IPE analysis results as a new input variable, in addition to existing input variables, to improve the prediction accuracy of these models. Three types of neural networks were adopted according to the development of artificial intelligence models: artificial neural networks (ANNs), long short-term memory networks (LSTMs), and Transformer models. The results indicate that all models using the proposed prediction method, which includes the IPE analysis result, outperformed those using the traditional prediction method. The Transformer & IPE model shows the best performance in prediction accuracy of the follower acceleration output; the statistically significant percentage improvements were 2.04%, 1.42%, 1.22%, and 2.62% for RMSE, MAE, MASE, and R^2, in that order. Furthermore, the results indicate that all models using the proposed prediction method outperformed the benchmarking Intelligent Driver Model (IDM) for the follower acceleration output.

Keywords: vehicle trajectory; improved permutation entropy; AI-based car-following models

Check for updates

Academic Editor: Cosimo Nardi

Received: 11 March 2026
Revised: 15 April 2026
Accepted: 21 April 2026
Published: 25 April 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.

1. Introduction

The car-following modeling framework aims to introduce a more realistic description of car-following behavior in different driving cases to enhance traffic safety and better understanding of many confusing traffic flow phenomena [1]. The complex and nonlinear nature of car-following behavior necessitates the development of intelligent algorithms to describe, model, and predict this phenomenon. Given the importance of car-following models in traffic flow analysis and because of their development over the past seventy years, studies have divided these models into two broad groups: the first group is mathematical car-following models, and the second is artificial intelligence-based car-following models [2–5].

Mathematical car-following models use mathematical expressions to explain car-following behavior based on kinematic principles and clear physical interpretations [2,4], such as the General Motors (GM) model (Cazis et al., 1961) [6], the Newell model

(Newell, 1961) [7], the Gipps model (Gipps, 1981) [8], the Intelligent Driver Model (IDM) (Treiber et al., 2000) [9], and the Full Velocity Difference (FVD) model (Jiang et al., 2001) [10]. Due to development in machine learning and deep learning, as well as the availability of large-scale data, the second group of artificial intelligence-based car-following models has recently emerged. These models can explore relationships between data, thereby discovering driving patterns and, subsequently, accurately predicting car-following behaviors [11]. Kehtarnavaz et al. (1998) [12] trained two-time delay backpropagation neural networks according to the collection of data, one for speed and the other for steering control; Panwai and Dia (2007) [13] used an ANN car-following model for mapping perceptions to actions; Khodayari et al. (2012) [14] established an ANN model with relative speed, relative distance, follower velocity, and instantaneous reaction delay as input variables to predict the following vehicle's behavior; Zhou et al. (2017) [15] used RNN (Recurrent Neural Network) with relative speed, relative distance, and follower velocity to predict traffic oscillation; Huang et al. (2018) [16] applied the LSTM model with relative speed, relative distance, and follower velocity to capture realistic traffic flow characteristics; Fangyu Wu & Daniel B. Work (2018) [17] used ANN with different network architectures and different activation functions to predict car-following models for traditional vehicles (TVs) for stop-and-go waves. The input variables were relative velocity, relative distance, and velocity of the follower vehicle; Ma and Qu (2020) [18] used a seq2seq LSTM model with relative speed, relative distance, and follower velocity to predict multiple-step acceleration/deceleration; Colombaroni et al. (2021) [19] used a feed-forward neural network (FF) and LSTM network to model car-following behavior for traditional vehicles (TVs). The input variables were relative speed, relative distance, and follower velocity to predict the acceleration or the velocity of the follower vehicle. They use reaction time as a further input variable, and they consider the reaction time as reaction duration and reaction lag; Zhou et al. (2022) [20] developed a transfer learning-based LSTM car-following model to handle the lack of adaptive cruise control (ACC) driving data; Qu et al. (2022) [21] used CNN-BiLSTM-Attention to establish a data-driven car-following model for connected and autonomous vehicles (CVs and AVs) and compared it with LSTM, GRU, and BiLSTM. The input variables included relative velocity, relative distance, and the velocity of the follower vehicle. The output variable was either the follower vehicle's acceleration, which could be used to predict its velocity and position, or just its position; Qin et al. (2023) [22] developed a CNN-LSTM car-following model with relative speed, relative distance, and follower velocity as input variables to predict the traffic flow behavior; Adewale & Lee (2024) [23] developed a PT-LSTM car-following model to predict the velocity of the following vehicle in autonomous and human-driven mixed traffic. The input variables were relative speed, relative distance, and follower velocity; Wang et al. (2025) [24] proposed a human-like car-following model based on deep deterministic policy gradient (DDPG) and neural network architecture with three input variables (relative speed, relative distance, and follower velocity) to predict the follower acceleration. The proposed model achieves superior anthropomorphic performance compared to IDM and ANN. Li et al. (2025) [25] proposed the Rational Artificial Intelligence Car-Following model Enhanced by Reality (RACER) and a deep learning algorithm that integrates Rational Driving Constraints (RDCs) into its loss function to safely predict for ACC vehicles. An LSTM network architecture with three input variables (relative speed, relative distance, and follower velocity) based on ACC vehicle experiments was used to predict the follower acceleration. Liu et al. (2025) [26] proposed a Phase-Aware AI (PAAI) car-following model that is designed specifically for electric vehicles (EVs) with ACC. The PAAI model integrated traditional car-following frameworks with AI corrections. For the baseline AI, they employed sequential neural networks with attention and phase-aware add-speed recognition. Three input variables (relative speed, relative distance,

and follower velocity) are used to predict follower acceleration. They found that the PAAI model outperforms base models (IDM and OVRV) and pure AI baselines. The integration of AI within car-following modeling represents not only methodological evolution but also a pathway to a more realistic simulation of mixed traffic systems [15].

Since vehicle trajectories are time series that sometimes contain similar values (for example, at a constant speed in free flow or during a stop-and-go situation) and sometimes contain dissimilar but close values, to distinguish between these readings, we need a powerful tool that accurately senses the differences that occur in vehicle trajectory data. Entropy tools are often used to test or measure how complicated a time series is according to a scale ranging from 0 to 1, where 0 indicates a regular time series and 1 indicates a random and irregular one [27,28]. Entropy is a powerful tool for the analysis of time series, as it allows the probability distributions of the possible states of a system and, therefore, the information encoded in it to be described [29]. Liu et al. [30] used the general formula for entropy; they applied the entropy to the distribution of vehicle space-headway on the road to estimate the dynamic traffic flow. The experiment was conducted under free-flow conditions on a one-lane road, using VISSIM simulation software to analyze different traffic volumes ranging from 800 to 1400 vehicles per hour. The results were that when the volume increases, entropy decreases because the vehicle spacing in the platoon tends to be shorter and uniform due to the large volume. Lyu et al. [31] used four types of entropy (Shannon entropy, steering wheel entropy, approximate entropy, and sample entropy), which applied to the time-headway data during car-following with fixed and different moving time windows. They found that not all entropy measures are appropriate in terms of car-following when measuring uncertainty in time-headway data. However, the researchers indicate that sample entropy and approximate entropy appear to be the most suitable measures, particularly when using a moving time window. Zhang et al. [32] used Fuzzy Permutation Entropy (FPE) to investigate the complexity of traffic evolution based on simulated time headway during different states of traffic congestion (Free Flow (FF), Synchronized Flow (SF), and both coexisting), and the data were generated from a cellular automation model based on the improved brake light rule (IBLR) model. Liu et al. [33] used Information Entropy (IE) for traffic flow analysis. They used NGSIM and a simulation dataset, and they found a linear relationship between speed (IE) with space-mean-speed and space-headway (IE) with density for traffic flow analysis. L. Tang [34] used information entropy to calculate the internal correlation among different indexes that represent driving behavior (e.g., speeding, rapid acceleration, sudden braking, etc.). Thereafter, index weights are derived from the information entropy results to establish a model for predicting safety risks in road traffic behaviors. Baldini et al. [35] used seven types of entropy measures (Shannon entropy, Rényi entropy, sample entropy, approximate entropy, permutation entropy, and dispersion entropy) to propose an intrusion detection system (IDS) that detects cyberattacks targeting the automotive in-vehicle Controller Area Network bus (CAN-bus) by analyzing the CAN-bus message payloads. They found that entropy measures with a sliding window approach can detect attacks in a time-efficient way and with high accuracy, particularly with approximate entropy. Guo et al. [36] found that applying information entropy to the flow data for urban tunnel sections can characterize traffic congestion effectively. Higher values of entropy for peak hours refer to greater congestion during peak times. Cui et al. [37] proposed a multiscale symbolic dynamic entropy method to analyze traffic flow complexity for weekdays and weekends. They found it effectively evaluates traffic flow complexity as compared to multiscale sample entropy and multiscale modified sample entropy. Ye et al. [38] proposed the velocity entropy method as a macroscopic safety indicator for road segments to quantify the overall safety status using real-time vehicle velocity data. Zhou et al. [39] used improved permutation entropy and Poincare mapping

to propose a threshold determination method for the Duffing chaotic oscillator detection system, which was used for weak signal detection. Zhou et al. [40] proposed an active detection method combining short-time Fourier transform and improved permutation entropy to detect a small underwater target and address the challenge of detecting weak target echoes in strong reverberation backgrounds.

From the foregoing, we observe that all research relying on car-following models did not consider the use of entropy in analyzing data and incorporating them as a new variable into those models to improve prediction accuracy and reduce error. Similarly, some research employing different types of entropy did not consider integrating entropy analysis into the study of car-following models at the micro level, which could enhance the understanding of driver behavior and improve the accuracy of traffic flow predictions.

We should know that there are many types of entropies based on Shannon's form, which is a mathematical concept used to measure information, that are widely used in various disciplines [28,32,41].

Compared with other entropy algorithms, Permutation Entropy (PE) [42] is famous and widely used for complexity series analysis [43]. It combines the concepts of entropy and symbolic dynamics to create a new measure of complexity; it is based on computing the Shannon entropy of the relative frequency of all the ordinal patterns found in a time series [44]. Chen et al. (2019) [45] proposed the IPE based on PE, which combines some advantages of previous PE modifications. They compared several tools, including PE, Multiscale Permutation Entropy (MPE), Amplitude-Aware Permutation Entropy (AAPE), and Weighted-Permutation Entropy (WPE), and found that the new IPE in the field of entropy detects spiky features more effectively. Furthermore, we conducted research that provided sufficient theoretical and experimental justification for determining the most effective type of PE or IPE for analyzing vehicle trajectories in car-following behavior and identifying the best variable from vehicle trajectory data; the results indicated that IPE outperformed PE in analyzing all traffic-related variables, with the local position variable being the most effective [46].

Most studies regarding the use of entropy in the field of transportation are limited to analyzing traffic conditions at the macro level, with hardly any studies focusing on car-following at the micro level. Some of these studies rely on hypothetical data in their analysis. Additionally, there is a lack of research on the application of entropy to intelligent transportation systems (ITSs) to develop car-following models for traditional vehicles (TVs), connected vehicles (CVs), autonomous vehicles (AVs), or a combination of these, as well as simulation programs. Furthermore, entropy tools are often applied to variables such as velocity, acceleration, space headway, and time headway, and their application to local position data of vehicle trajectory has not been previously addressed.

Therefore, this study represents the first large-scale investigation of its kind into the ability of IPE to analyze lead vehicle position data, aiming to understand individual vehicle behavior during car-following and to introduce a new prediction method for developing artificial intelligence-based car-following models that enhance their predictive accuracy.

The novelty of this study is that it proposes a new prediction method for developing artificial intelligence-based car-following models. It is using IPE for the first time to analyze lead vehicle trajectory (local position data) and incorporating the analysis results as a new input variable alongside existing input variables to enhance the prediction accuracy of these models.

We summarize this paper's contributions as follows:

- Applying a new prediction method by using the IPE to analyze lead vehicle position data and incorporating the analysis results to develop artificial intelligence-based car-following models.

- Using and comparing four different models (ANN, LSTM, Transformer, and IDM) to examine the prediction method for follower acceleration forecasting.
- Different tests were conducted to provide a comprehensive evaluation of the models' performance by using four evaluation metrics (RMSE, MAE, MASE, and R^2), statistical testing, ablation experiments, and feature contribution analysis.
- Offering helpful information regarding the ability of the models to predict the follower vehicle acceleration.

The rest of the paper is organized as follows: Section 2 presents the Improved Permutation Entropy (IPE) algorithm and its analysis of real vehicle trajectory data. Section 3 introduces the model formulation (including prediction methods, data processing, and evaluation methods). Section 4 presents prediction models (ANN, LSTM, Transformer, and IDM). In Section 5, the results and discussion are presented. Section 6 provides the conclusions.

2. Materials and Methods

2.1. IPE Algorithm and Its Parameter Setting

IPE [45] has been used in this study to quantify and measure the complexity of the time series (analyzing the trajectory's data for vehicles during car-following behaviors).

The normalized IPE is written as:

$$IPE(D, \tau, L) = -\frac{1}{\ln(L^D)} \sum_{l=1}^{h} P_l \ln P_l, \quad h \leq L^D \tag{1}$$

where L is the number of discretization levels, D is the embedding dimension, P_l is the probability of permutation pattern π_l, $h \leq L^D$, and $\ln(L^D)$ is the maximum value of the IPE analysis results, which is only reached when the patterns have a uniform distribution; see Appendix A.

There are some parameters whose values must be predetermined for running the IPE algorithm for any data used in this study. These parameters include τ (time delay); it controls the number of elements of N to be selected in the entropy calculation. For example, if $\tau = 1$, all the elements in the time series are selected one by one. If $\tau = 2$, the first, third, fifth, etc., elements are selected, and the second, fourth, sixth, etc., elements are omitted [42]. In this study, $\tau = 1$ was set to ensure that all time series data were included in the calculations and for structural preservation.

The other parameters include embedding dimension (D) and level of discretization (L); for example, when $D = 3$ (the minimum value proposed by Bandt and Pompe (2002), $3 \leq D \leq 7$) [42], this value means we have three numbers $\{0, 1, 2\}$ that can be generated inside the IPE algorithm. This allows for 27 patterns to be formed to arrange these numbers, as shown in Table A1; see Appendix A. Any L^D (L^D represents the possible permutation patterns in the IPE algorithm) value in Table 1 less than 27 indicates that some of the time series data have lost their true representation in the patterns listed in Table A1. Conversely, any L^D value greater than 27 indicates the repeated use of one of the patterns in Table A1 to represent different data from the time series. Furthermore, the length of the time series N is crucially related to the values of D and L. According to Refs. [42,45,47], $N > L^D$ is necessary to achieve reliable IPE measurements. If we assume $D = L = 4$, this means $L^D = 256$. Since each element in the time series N is 0.1 s, the minimum length of the N time series would be greater than 25.6 s, $N > L^D$. This time is too long to obtain the first test result for the N time series in the field of car-following.

Table 1. L^D values based on D and L parameters.

L \ D	3	4
2	8	16
3	27	81
4	64	256

In this study, $D = L = 3$ was adopted to achieve a true representation of the time series data with IPE analysis results, and $N = 30$ was chosen, which represents 3 s of distance traveled while also fulfilling the condition $N > L^D$.

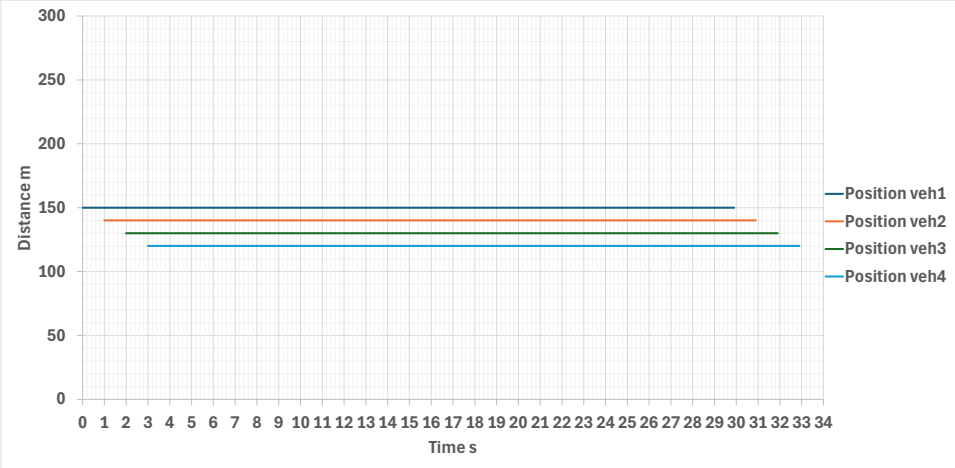
Finally, SL represents the total length of the time series, which corresponds to the duration for examining sample data.

Since $N = 30$, a 3 s delay occurs before obtaining the first IPE analysis result (initial processing event) when starting the IPE calculation for the SL time series. Therefore, a 3 s moving time window was used when testing the SL time series for each trajectory, confining the delay to the beginning of the test only; see Appendix A.

This study aims to use the IPE to analyze lead vehicle position data during car-following processes. Therefore, to understand the results of the analysis, four hypothetical cases were considered regarding the movement of vehicles.

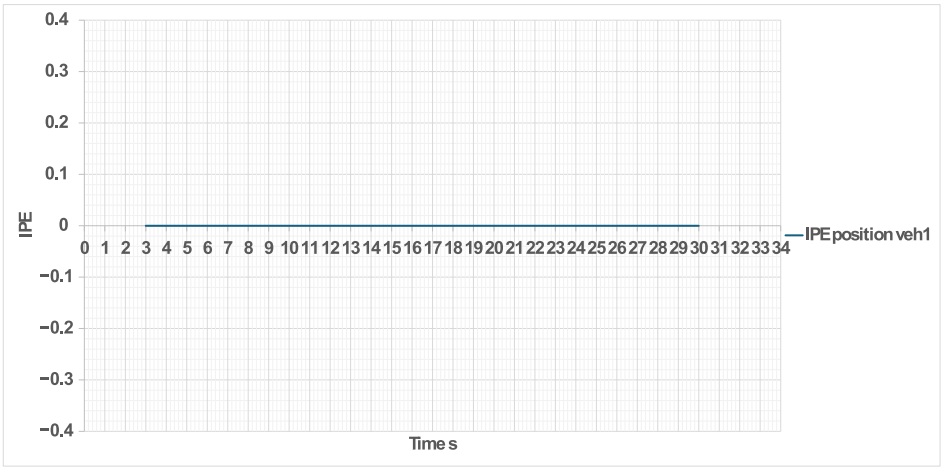
2.1.1. First Hypothetical Case: Stopping Without Movement

Consider a road segment in which a vehicle (veh1) remains stopped at a station located 150 m from the starting point for a duration exceeding 3 s, as shown in Figure 1a. Then the time series for this vehicle, with $N = 30$, will be represented as $X = [150, 150, \ldots, 150]_{i=1}^{N=30}$. Based on the entropy definition and the specified IPE parameters, the IPE analysis result for this series will be zero (due to completely regular behavior). Furthermore, when computing the IPE over a sample length of $SL = 300$, the IPE analysis results remain zero throughout, as shown in Figure 1b. For more details about the calculation of IPE, see Appendix A, Figure A1, and Refs. [45,48].



(**a**) Vehicle trajectories when stopped without movement.

Figure 1. *Cont.*

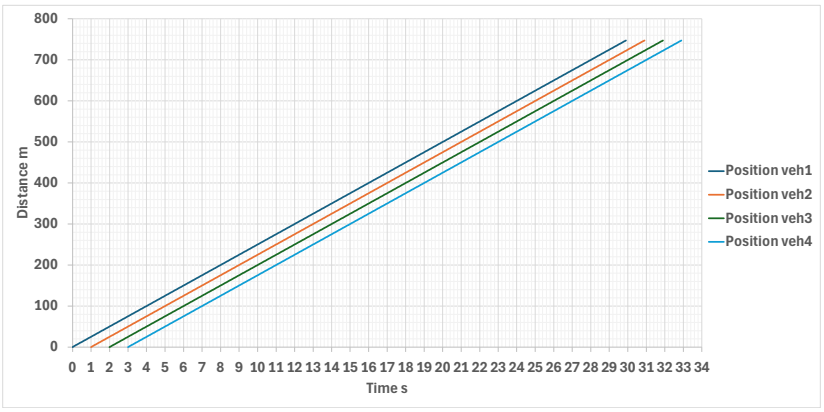


(**b**) IPE analysis results for trajectory of veh1.

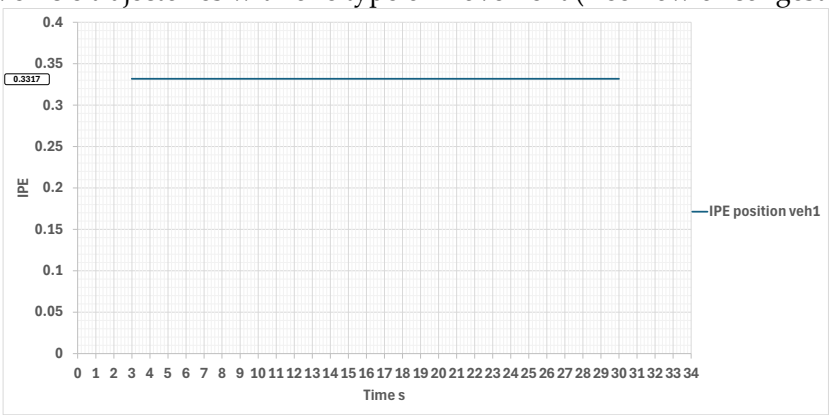
Figure 1. IPE applied to trajectories without movement.

2.1.2. Second Hypothetical Case: One Type of Movement (Free Flow or Congestion) Without Stopping

Consider a road segment on which a vehicle (veh1) starts from station 0 at a constant speed, covering a distance of 2.5 m every 0.1 s, as shown in Figure 2a. The time series for this vehicle, with $N = 30$, will be represented as $X = [2.5,\ 5,\ \ldots,\ 75]_{i=1}^{N=30}$.



(**a**) Vehicle trajectories with one type of movement (free flow or congestion).



(**b**) IPE analysis results for trajectory of veh1.

Figure 2. IPE applied to trajectories with one type of movement (free flow or congestion).

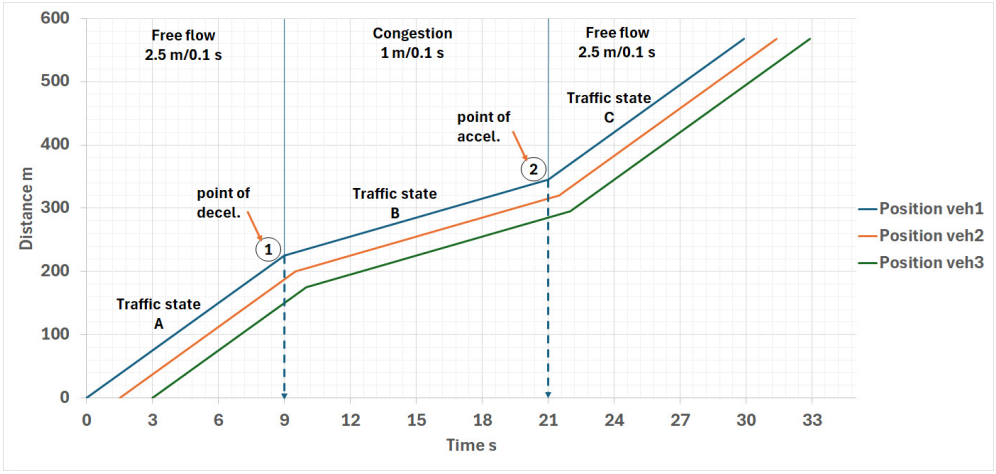
Based on the entropy definition and the specified IPE parameters, the IPE analysis result for this N-element series will be equal to 0.3317. Continuing to calculate the IPE for

$SL = 300$ will also yield 0.3317 for the IPE analysis results, as shown in Figure 2b. For more details about the calculation of IPE see Appendix A, Figure A2, and Refs. [45,48].

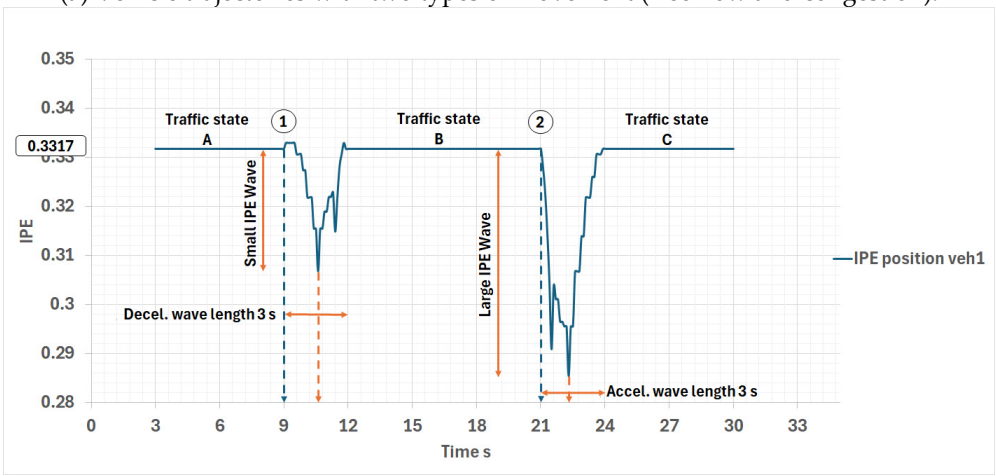
2.1.3. Third Hypothetical Case: Two Types of Movement (Free Flow and Congestion) Without Stopping

Depending on the concepts and information given in Sections 2.1.1 and 2.1.2 above, we can assume the following:

Consider a road segment on which a vehicle (veh1) starts from station 0 at a constant speed, and there are two types of movements on this road: free flow and congestion, as shown in Figure 3a. In free flow, the vehicle travels 2.5 m every 0.1 s, and in congestion, it travels 1 m every 0.1 s. The duration for examining sample data is 30 s ($SL = 300$). Therefore, based on Section 2.1.2 and the specified IPE parameters, the IPE analysis results will be constant at 0.3317 for traffic states A, B, and C during the vehicle cruise for both types of movement, as illustrated in Figure 3b. However, at transition points ① and ② (which represent deceleration and acceleration, respectively), the IPE analysis results will vary depending on the speed or displacement of vehicles before and after these points.



(**a**) Vehicle trajectories with two types of movement (free flow and congestion).



(**b**) IPE analysis results for trajectory of veh1.

Figure 3. IPE applied to trajectories with two types of movement (free flow and congestion).

As shown in Figure 3b, we can see there are small and large IPE waves; the values of the IPE analysis results for both do not equal zero. The small wave represents the transition from free flow to congestion (point ① of deceleration, Figure 3a), while the large wave represents the transition from congestion to free flow (point ② of acceleration, Figure 3a).

The time length of small or large IPE waves is equal to 3 s because $N = 30$ and the moving time window equals 3 s (Section 2.1).

From Figure 3b, we can conclude that the time of the minimum IPE analysis result of the small wave coincides approximately with the time of the deceleration point ① in Figure 3a, and the time of the minimum IPE analysis result of the large wave coincides approximately with the time of the acceleration point ② in Figure 3a; however, this transition occurs only after approximately 1.5 s have passed because each IPE analysis result does not appear until 3 s after the IPE calculation for the time series begins. This phenomenon was previously mentioned in Section 2.1 (since $N = 30$, the value indicates a time lag or delay of 3 s before obtaining the first IPE at the start of the IPE calculation for the time series).

2.1.4. Fourth Hypothetical Case: All Types of Movements (Free Flow, Congestion, Stop, and Free Flow)

Based on the concepts and information given in Sections 2.1.1–2.1.3 above, we can assume the following:

Consider a road segment on which a vehicle (veh1) starts from station 0 and travels with all types of movements on this road: free flow, congestion, and stopping, as shown in Figure 4a. In free flow, the vehicle travels 2.5 m every 0.1 s, and in congestion, it travels 1 m every 0.1 s. The duration for examining sample data is 30 s (SL = 300). Therefore, based on the specified IPE parameters, the IPE analysis results for traffic states A, B, and D will be constant at 0.3317 according to Section 2.1.2, as illustrated in Figure 4b; for traffic state C, the results are zero according to Section 2.1.1, as illustrated in Figure 4b; while for transition points (①, ②, and ③), the results vary according to the speed or displacement of the vehicle before and after these points, as depicted in Figure 4a,b.

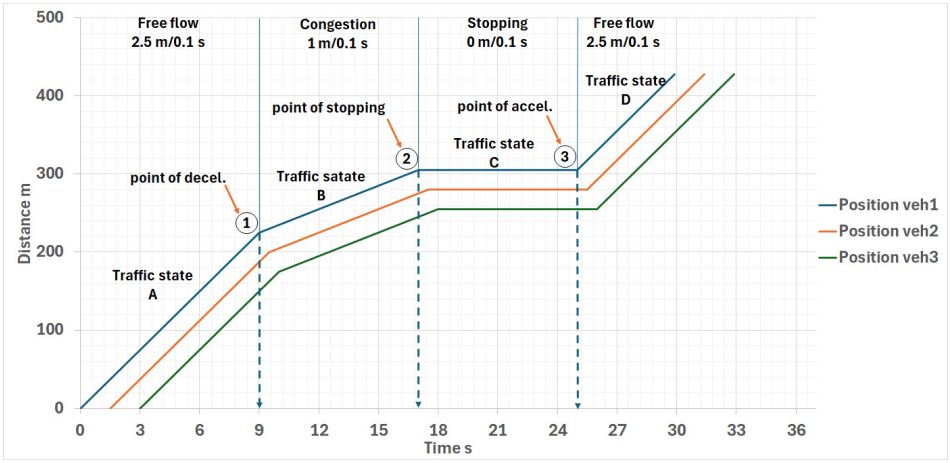
As shown in Figure 4b, we can see there is a small IPE wave representing the deceleration wave, and it is exactly like the small IPE wave mentioned in Section 2.1.3., starting at point ① in Figure 3b. Furthermore, we can see two other waves. The stopping IPE wave represents the transition point ② from congestion to stopping in Figure 4a, while the other is the acceleration IPE wave, and it represents the transition point ③ from stopping to free flow in Figure 4a. The time length of a stopping IPE wave or acceleration IPE wave equals 3 s because $N = 30$ and the moving time window is equal to 3 s (Section 2.1).

The time of the minimum value of the IPE analysis results in the stopping wave in Figure 4b represents the time of the end of the wave, and it coincides with the time of the transition point ② from congested flow to stop, but after 3 s, as shown in Figure 4a. The reason for this, as mentioned in Section 2.1.3., is that each result of the IPE analysis results does not appear until 3 s after the IPE calculation for the time series begins. On the other hand, at the next wave, the time of the minimum value of IPE analysis results in the acceleration wave in Figure 4b represents the time of the starting of the wave, and it coincides with the time of the transition point ③ from stopping to free flow, as shown in Figure 4a. This is because the IPE analysis results remain constant when the time series values are constant, and any subsequent change in these values will indicate the start of the change in the newly calculated IPE analysis results.

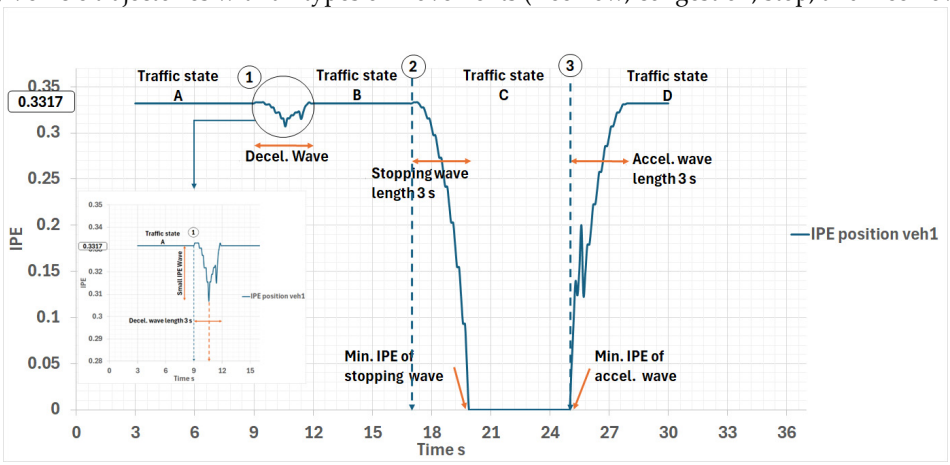
We can conclude that the time of the end of the stopping wave in Figure 4b minus 3 s coincides with the time of the stopping point ② in Figure 4a. The time of starting of the acceleration wave in Figure 4b exactly coincides with the time of the acceleration point ③ in Figure 4a (on the condition that acceleration starts from zero).

Based on these hypothetical cases, it can be concluded theoretically that the IPE can convert vehicle position data from an unlimited range ($0 \rightarrow \infty$ m, unlimited travel distance during continuous movement) to a limited range ($0 \rightarrow 0.3317$, where 0 means stopping and 0.3317 means uniform movement). Between stop and movement, either acceleration or deceleration occurs, which corresponds to IPE analysis results that range from 0 to

0.3317 (IPE is dimensionless). Thus, this limited range is like that of variables such as speed. For example, speed reflects the vehicle's movement behavior (stationary, slow, quick, very fast, etc.) based on its speed value. The IPE analysis result also reflects the vehicle's movement behavior, but in a different way (stationary, unstable, or uniform), regardless of the vehicle's speed value. In other words, the speed might be slow, but the IPE analysis results would be uniform because the acceleration is zero. Conversely, the speed might be high, but the IPE analysis results would be unstable due to variations in deceleration or acceleration depending on the movement of the vehicle ahead. This description applies to an individual vehicle; however, when IPE is applied to a group of vehicles on a specific road, the traffic movement or stability of flow on that road can be characterized as "stationary," "unstable," or "uniform," regardless of the speed on that road.



(**a**) Vehicle trajectories with all types of movements (free flow, congestion, stop, and free flow).



(**b**) IPE analysis results for trajectory of veh1. The IPE wave inside the circle at point ① is exactly like the small IPE wave in Figure 3b.

Figure 4. IPE applied to trajectories with all types of movements (free flow, congestion, stop, and free flow).

Position data in vehicle trajectories are smoother than speed or acceleration data. Hence, converting position data to IPE provides richer data content for identifying traffic states.

2.2. Data Preperation for IPE Applications

Real car-following data are adopted in this empirical study, and they are extracted from the trajectory dataset of the Next Generation Simulation (NGSIM) project [49]. The data were collected on 15 June 2005, during a 15 min observation period (7:50 a.m.–8:05 a.m.)

on a segment of U.S. Highway 101, the Hollywood Freeway, in Los Angeles, California. These data were made using eight cameras mounted on a 36-story building, 10 Universal City Plaza, next to the Hollywood Freeway U.S.-101. On a road section of 640 m, the data were collected in 0.1 s intervals. The freeway segment consists of six lanes. In this empirical study, we focused on traffic flow patterns based on the car-following behaviors in the first lane, which is termed lane 1.

We prepared a dataset of one group of trajectories (G1), as shown in Figure 5 and Table 2, for the purpose of illustrating how to apply IPE to analyze real vehicle trajectory data.

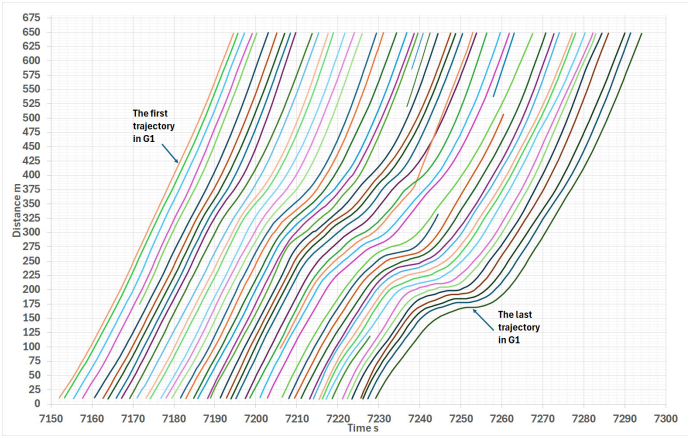


Figure 5. Selected group of vehicle trajectories in G1.

Table 2. Detail of selected trajectories in G1.

Group ID	Number of Vehicles	First Veh_ID in Group	Last Veh_ID in Group
G1	48	719	1058

2.3. Applying IPE to Analyze Real Vehicle Trajectory Data

After implementing IPE to the vehicle trajectories in G1 (position data) and for explanation, we randomly selected the trajectory of vehicle No. 966, as shown in Figure 6, as the blue trajectory. This trajectory illustrates the deceleration and acceleration points. The green line illustrates the IPE analysis results of the vehicle trajectory, displaying two waves. According to Section 2.1.3, a small IPE wave indicates a deceleration point, and a large IPE wave indicates an acceleration point.

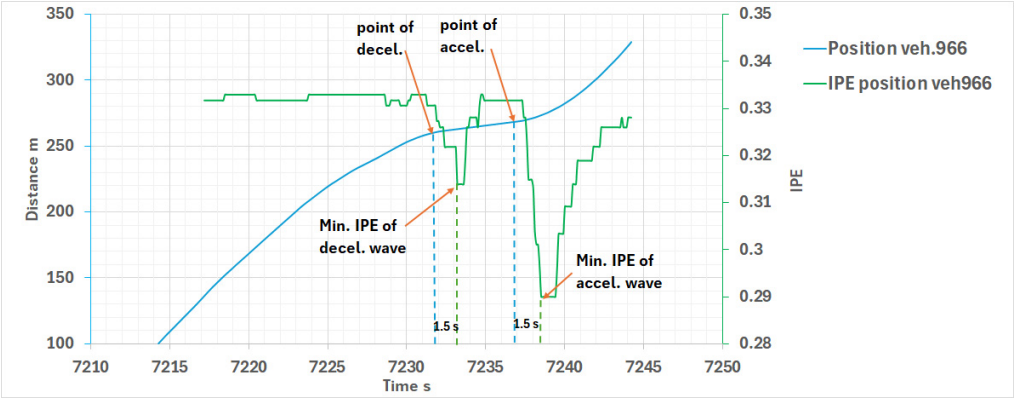


Figure 6. Analysis of (position veh.966) trajectory by using the IPE.

Figure 7 presents the IPE analysis results for all vehicle trajectories in G1. When comparing the IPE analysis results to the baseline of 0.3317 mentioned in Section 2.1.2, we observe that the difference in IPE analysis results for the first 25 vehicles is very close to zero, indicating a streamlined movement that is compatible with the type of flow for these vehicles illustrated in Figure 5. The difference in IPE analysis results for the remaining vehicles increases, which refers to unstable flow that is compatible with the congestion illustrated in Figure 5. These IPE analysis results indicate that the movement in G1 transitions from free flow to congestion without stopping because the IPE analysis results are not equal to zero, and this pattern is completely consistent with the type of flow in G1, as shown in Figure 5.

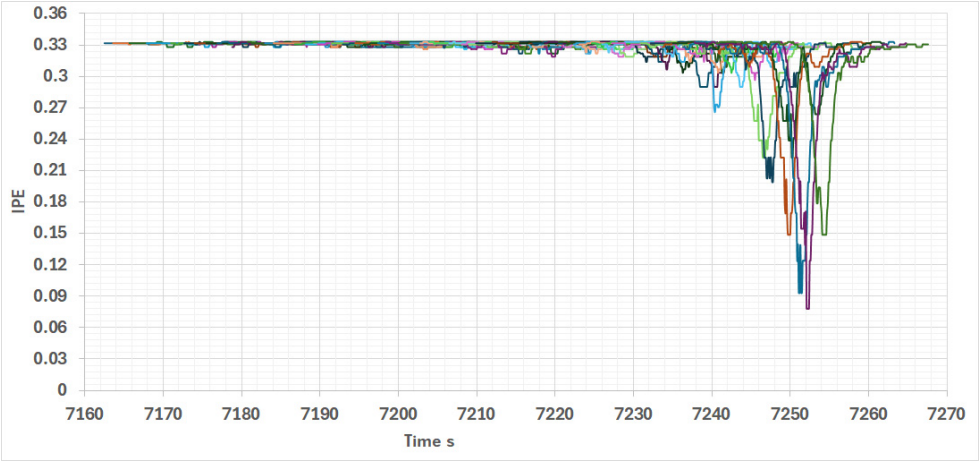


Figure 7. Analysis of all trajectories in G1 by using the IPE.

According to Section 2.1.2, the IPE identified that the value of 0.3317 refers to streamlined movement, and any deviation from this baseline (0.3317) indicates a variation in velocity that is proportional to it. For example, the first vehicle, No. 719 in Figure 5, with an average speed of 60 km/h, represents streamlined movement, and its IPE analysis results were very close to 0.3317, as shown in Figure 8. While the last vehicle, No. 1058 in Figure 5, with an average speed of 18 km/h, presents irregular movement, its IPE analysis results were significantly different from 0.3317, as shown in Figure 8. In other words, any change in the IPE analysis results for the lead vehicle will directly affect the following vehicle, and this is precisely what car-following theory represents.

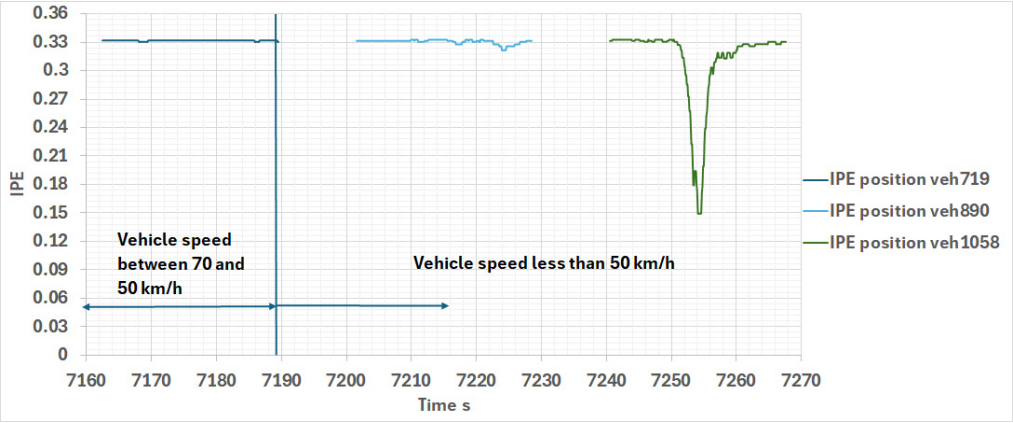


Figure 8. IPE analysis results for specific trajectories according to the speed of the vehicle.

To maximize the benefit derived from the IPE analysis results, we proposed a new prediction method to develop an artificial intelligence-based car-following model by using

the calculated IPE analysis results as a new input variable. The following sections explain the proposed methods.

3. Model Formulation

3.1. Prediction Methods

This study adopted two types of prediction methods based on the explanations provided in the literature and the concepts of the GHR modeling [6]. First, we selected the velocity of the follower vehicle (V_n), relative velocity (Δv), and relative distance (Δx) at time t as input variables to predict the acceleration of the follower vehicle (a_n) at time $t + \Delta t$, as shown in Equation (2). These selections represent the traditional prediction method currently in use and are adopted for comparison purposes later. Second, in addition to the variables in the first prediction method, we added the IPE analysis results of the lead vehicle's position data at time t as a new input variable to predict the acceleration of the follower vehicle (a_n) at time $t + \Delta t$, as shown in Equation (3). This option represents the proposed prediction method to develop artificial intelligence-based car-following models using the calculated IPE analysis results as a new input variable.

$$\begin{bmatrix} \Delta v_{(t)} \\ \Delta x_{(t)} \\ V_{n(t)} \end{bmatrix} \longrightarrow a_{n(t+\Delta t)} \tag{2}$$

$$\begin{bmatrix} \Delta v_{(t)} \\ \Delta x_{(t)} \\ V_{n(t)} \\ IPE_{(t)} \end{bmatrix} \longrightarrow a_{n(t+\Delta t)} \tag{3}$$

where

$\Delta v_{(t)}$: relative velocity (m/s) between the leader and the follower at current time t.
$\Delta x_{(t)}$: relative distance (m) between the leader and the follower at current time t.
$V_{n(t)}$: velocity of the follower vehicle (m/s) at current time t.
$IPE_{(t)}$: IPE analysis results of the leader vehicle's position data at current time t.
$a_{n(t+\Delta t)}$: acceleration of the follower vehicle (m/s^2) at the next time step $t + \Delta t$.

The prediction methods output the acceleration of the follower vehicle, so we calculated the follower velocity and follower position using Equations (4) and (5) below. They provide an additional measure to verify the stability of all the models used in this study.

$$V_{n(t+\Delta t)} = V_{n(t)} + a_{n(t+\Delta t)} \Delta t \tag{4}$$

$$X_{n(t+\Delta t)} = X_{n(t)} + V_{n(t)} \Delta t + \frac{1}{2} a_{n(t+\Delta t)} \Delta t^2 \tag{5}$$

where

V_n is the velocity of the follower vehicle n (m/s).
$a_{n(t+\Delta t)}$ is the acceleration of the follower vehicle n (m/s^2) at the next time step.
X_n is the position of the follower vehicle n (m).
(t) is the current time.
$(t + \Delta t)$ is the next time step.
$\Delta t = 0.1$ s is the time resolution of the NGSIM dataset and represents one time step.

3.2. Data Processing

As mentioned in Section 2.2, a real car-following dataset was adopted in this empirical study; it was extracted from the same trajectory dataset of the Next Generation Simulation (NGSIM) project.

A sample of 100 trajectories was selected based on the local position data (Local_Y) of vehicle trajectories in NGSIM and after removing abnormal trajectories. This sample of trajectories represents the initial trajectories of the NGSIM data, and it was divided into 70% for training and 30% for testing. Each trajectory includes a time series that ensures its integrity.

The trajectory data seemed to be unfiltered and exhibited some noise, so a moving average filter was applied to all trajectories for a duration of 3 s. This period was set to match the IPE calculation period mentioned in Section 2.1.

The selected trajectory data were normalized to eliminate the influence of dimensions, following the formula presented in Equation (6).

$$y^* = \frac{y - y_{min}}{y_{max} - y_{min}} \tag{6}$$

where y_{max} and y_{min} present the maximum and minimum value, respectively, in the original data.

3.3. Evaluation Methods

Four types of evaluation metrics were used to compare the competing models in terms of their prediction performance. Root Mean Square Error (RMSE) measures the average difference between actual and predicted values, as depicted in Equation (7), where a lower RMSE indicates better predictive accuracy. Mean Absolute Error (MAE), as presented in Equation (8), is the average of the absolute differences between the predicted and actual values. A lower MAE is better. Mean Absolute Scaled Error (MASE), as presented in Equation (9), is calculated by scaling the MAE of the model against the MAE of a naïve baseline forecast, with a lower MASE indicating better performance. The coefficient of determination (R^2) was calculated for further comparison and explains model fit quality (10). It ranges from 0 to 1; 1 means the regression model fits the data perfectly. Below are the equations used to calculate these metrics.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{7}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{8}$$

$$MASE = \frac{\frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|}{\frac{1}{n-1}\sum_{i=2}^{n}|y_i - y_{i-1}|} \tag{9}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}\left(y_i - \widetilde{y}_i\right)^2} \tag{10}$$

where n denotes the number of total evaluation data samples, y_i denotes the *i-th* true value, $\hat{y}_i$ is the predicted value, and $\widetilde{y}$ represents the mean value of the true values.

4. Prediction Models

In this study, three types of neural networks were adopted according to the growth of artificial intelligence models: artificial neural networks (ANNs), long short-term memory

networks (LSTMs), and Transformer models. The IDM was also used to evaluate the competence of the AI-based car-following models. Figure 9 shows the flowchart of the overall modeling process.

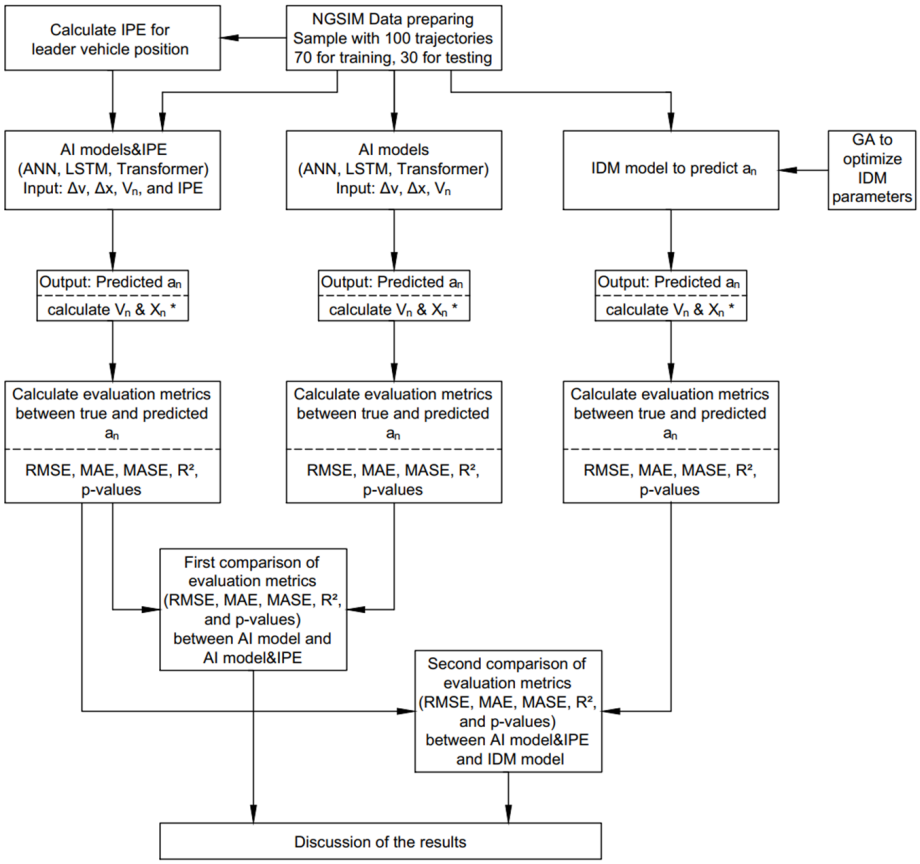


Figure 9. Flowchart of the overall modeling process. * Calculate the Vn and Xn based on the kinematic Equations (4) and (5), respectively.

4.1. ANNs

We adopted a generic form of ANN. The ANN model architecture included an input layer with different input variables according to the prediction methods, two hidden layers with five neurons each, and an output layer with one output variable, as shown in Table 3.

Table 3. Different input–output scenarios for ANN training.

Prediction Method	Input Layer	Network Size	Output Layer
1	3 neurons [Δv, Δx, V_n]	2 hidden layers with neurons each	1 neuron (follower acceleration)
2	4 neurons [Δv, Δx, V_n, IPE]		

Δv: relative velocity (m/s); Δx: relative distance (m); V_n: velocity of the follower vehicle (m/s); IPE: IPE analysis results of the leader vehicle position.

The backpropagation algorithm was applied to train the proposed ANN models. It computes the gradient of the loss function by propagating the prediction error backward through the network layers. In this study, the Scaled Conjugate Gradient (SCG) algorithm (trainscg) was used as the training function to update the network weights and biases [50]. SCG is efficient in terms of computation and works well with big datasets [50,51]. The training function minimizes the error between predicted and real outputs using gradients

obtained via backpropagation [51]. The ANN was implemented using MATLAB R2025b to train 70% of the dataset. A total of 15% of the training dataset was reserved for validation. The remaining 30% of the dataset was used to evaluate the ANN models. Table 4 presents the parameters of the ANN model that were used for both prediction methods.

Table 4. Architecture and parameters of the ANN models used for both prediction methods.

Parameter	Specification
Input layer size	4 input neurons
Hidden layers	2 hidden layers
Hidden layer 1	5 fully connected neurons
Hidden layer 2	5 fully connected neurons
Output layer	1 fully connected neuron
Activation function (hidden)	Nonlinear activation tansiq (default)
Training algorithm (optimization method)	Scaled conjugate gradient (trainscg)
Loss function	MSE
Epochs	1000 (default)
Performance goal	0 (default)
Minimum gradient	1×10^{-6}

4.2. LSTMs

The matter of gradient vanishing, a significant block for traditional recurrent neural networks, prompted Hochreiter and Schmidhuber in 1997 [52] to discover long short-term memory (LSTM) networks to address this challenge. They were a big step forward in deep learning because of their unique design and control mechanisms [53,54]. LSTM networks are an improved version of recurrent neural networks. They are designed to store information from past data for extended periods. Their design for processing sequential data also makes them adept at using past data to predict future events [55,56].

The core idea of LSTM networks is that they have various components whose structure is a sophisticated framework designed to store and manage information over long sequences. The cell state, the hidden state, and the gates are all these components. The network memory is the cell state, and it is designed to keep information from earlier time steps to avoid the problem of vanishing gradient. The hidden state is used for making predictions. Finally, there are three types of gates, each serving a specific purpose:

- Input gate: Using the sigmoid function to determine which new information from the current inputs will be used to update the cell state.
- Forget gate: Using the sigmoid function to determine which information from the previous cell state should be discarded.
- Output gate: This regulates which information from the network's memory will be output.

Figure 10 shows the structure of the LSTM network, including the connections between the gates, cell state, and hidden state.

In this study, the LSTM models were implemented using MATLAB to train 70% of the dataset. A total of 15% and 10% of the training dataset was reserved for validation and testing, respectively. The remaining 30% of the dataset was used to evaluate the LSTM models based on the types of prediction methods applied. Table 5 presents the parameters of the LSTM model that were used for both prediction methods, and Figure 11 shows the LSTM network architecture based on the proposed prediction methods.

Table 5. Architecture and parameters of the LSTM model used for both prediction methods.

Parameter	Specification
Input layer	Sequence input with 4 features
LSTM layer	20 hidden units
Fully connected layer 1	10 neurons
Output layer	1 fully connected neuron
Activation function	ReLU
Optimizer	RMSProp
Epochs	100
Learning rate	0.001

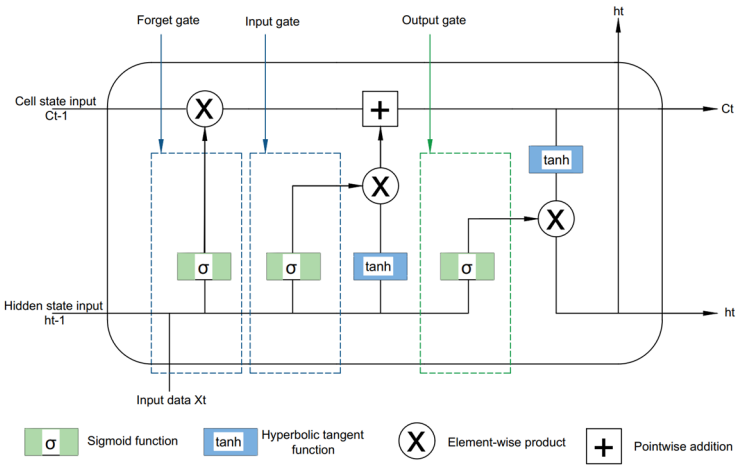


Figure 10. The structure of the LSTM network. For more information, see Refs. [53,54,57].

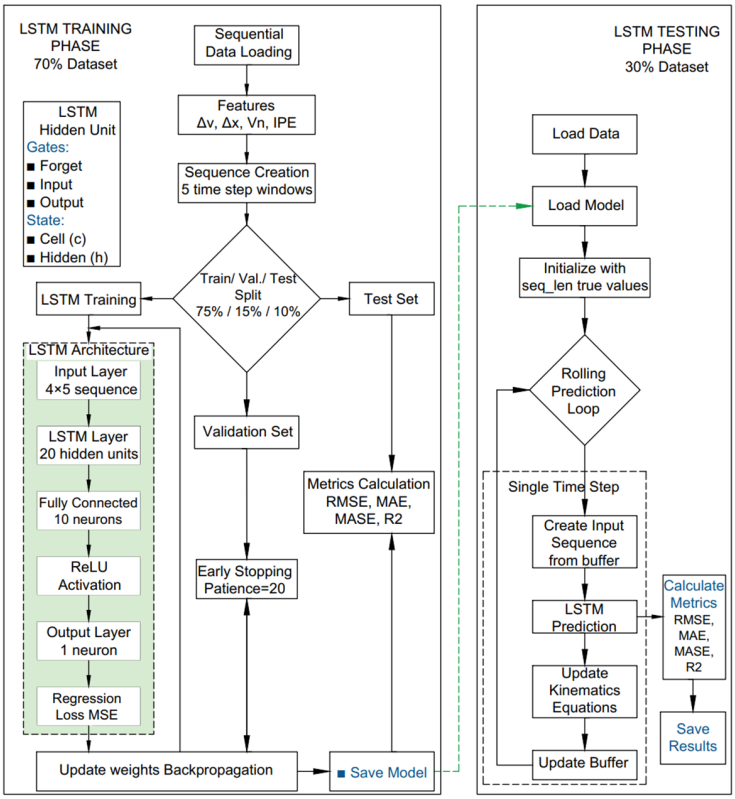


Figure 11. LSTM network architecture based on the proposed prediction methods.

4.3. Transformer

Transformer models are among the most powerful and newest types of neural network models [58,59]. They were first introduced in 2017 under the title of "Attention is All You Need" [60] and are now deemed a turning point in deep learning. The main feature of transformer models is their self-attention mechanism [60], which grants them the maximum ability to process sequential data and discover the relationships between parts of the input sequence. This feature enabled transformers to outperform other neural networks, like CNN (Convolution Neural Network) and RNN (Recurrent Neural Network). RNNs can process short sequences in a specific order because they receive the input sequence elements one by one. LSTM networks have mitigated this constraint [52]. However, transformers outperform in this regard due to their attentional mechanisms that can examine the entire sequence simultaneously and clearly understand long-range dependencies [61]. Furthermore, this feature allows for parallel processing, enabling the achievement of multiple computational steps at once instead of sequentially. Additionally, its ability to perform parallel training and capture long-range dependencies has garnered considerable attention, leading to its widespread use in various disciplines, such as language processing, computer vision [62–66], and time series prediction [67].

The operation of transformer models can be summarized in the following steps [60]:

1- Input embeddings.
2- Positional encoding.
3- Generating vectors.
4- Self-attention mechanisms.
5- Multi-head attention.
6- Layer normalization and residual connections.
7- Output layer.

In this study, the Transformer models were implemented using Python 3.14 to train 70% of the dataset. A total of 15% and 10% of the training dataset was reserved for validation and testing, respectively. The remaining 30% of the dataset was used to evaluate the Transformer models based on the types of prediction methods used. Table 6 presents the parameters of the Transformer model that were used for both prediction methods, and Figure 12 shows the Transformer network architecture based on the proposed prediction methods.

Table 6. Architecture and parameters of the Transformer model used for both prediction methods.

Parameter	Specification
Input embedding layer	Linear layer: 4 → d_model (64)
Embedding size (d_model)	64
Transformer encoder layers	3 layers
Self-attention	Multi-head attention with 4 heads
Feedforward dimension	128
Output layer	Linear layer: d_model (64) → 1
Activation function	ReLU (default)
Optimizer	Adam
Loss function	MSE
Learning rate	0.001
Epochs	40

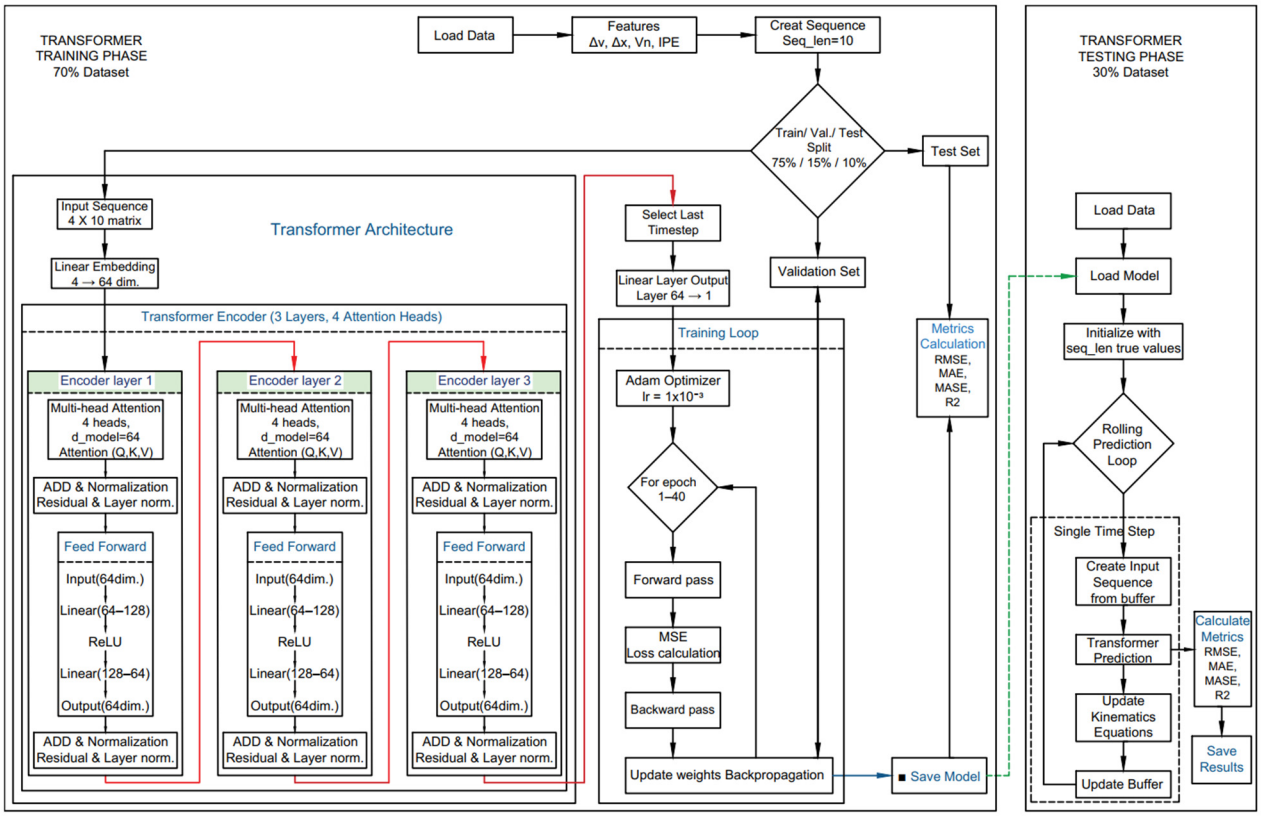


Figure 12. Transformer architecture network based on the proposed prediction methods.

4.4. IDM

Over the years, numerous classical mathematical car-following models have been proposed, contributing to our understanding of driver behavior and traffic flow. These models have produced varying levels of performance in terms of computational efficiency, accuracy, and realism [68]. Among these models, the Intelligent Driver Model (IDM) [9] is distinguished as the most widely used in various applications for understanding traffic conditions such as congestion, stop-and-go oscillations, and free flow. It is also distinguished by its ability to adjust acceleration based on factors such as relative speed, safety distance, and desired velocity. In this study, it was adopted for comparison with the selected artificial intelligence models (ANN, LSTM, and Transformer) to provide an additional benchmark for the accuracy of the conclusions.

With MATLAB software, the IDM was implemented using the dataset, with 70% for training and 30% for testing. Because the IDM has five parameters that need to be calibrated (a, b, S_o, T, and V_o), the Genetic Algorithm (GA) was used to calibrate and optimize these parameters and then identify the parameter set that minimized the error between the model output and the actual vehicle behavior data, as shown in Figure 13. The iterative process (generations) continued until the stopping criterion (when the average change in the fitness value was less than 1×10^{-4}) was satisfied. The IDM with the optimized parameter set was evaluated by Equation (11) using the remaining 30% test dataset.

$$a_{n(t+\Delta t)} = a\left[1 - \left(\frac{V_{n(t)}}{V_o}\right)^{\delta} - \left(\frac{S^*}{S_{(t)}}\right)^2\right] \tag{11}$$

where

$a_{n(t+\Delta t)}$ is the acceleration of the follower vehicle at the next time step, $\Delta t = 0.1$ s.
a is the maximum acceleration.

$V_{n(t)}$ is the speed of the follower vehicle at time step (t).
V_o is the desired velocity.
δ is the acceleration exponent (set to 4).
$S_{(t)}$ is the actual gap to the leading vehicle at time step (t).
$S_{(t)} = Leader\ Position - Follower\ Position - Leader\ Length.$
S^* is the desired minimum gap, calculated as:

$$S^* = S_o + V_{n(t)}T + \frac{V_{n(t)}\Delta V_{(t)}}{2\sqrt{ab}} \quad (12)$$

S_o is the minimum gap.
T is the desired time headway.
$\Delta V_{(t)} = V_{n(t)} - V_{n-1(t)}$ is the relative velocity between the leader and follower vehicles.
$V_{n-1(t)}$ is the speed of the lead vehicle at time step (t).
b is the comfortable deceleration.

After calibrating the IDM using the training dataset, the optimized parameter set was determined as follows: $a = 1.17\ m/s^2$, $b = 2.13\ m/s^2$, $S_o = 3.37$ m, $T = 0.99$ s, and $V_o = 26.78$ m/s.

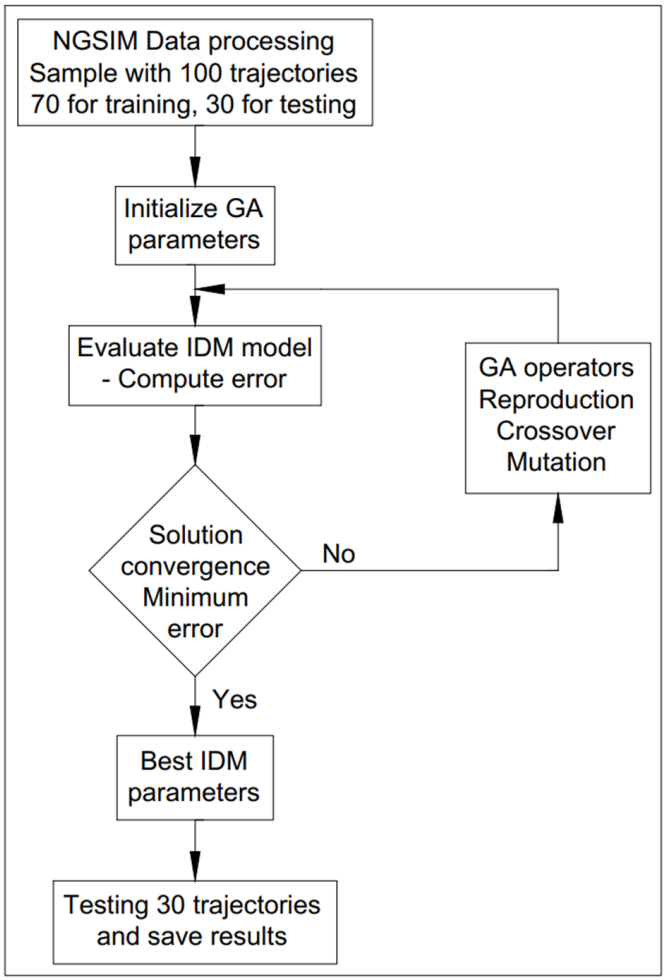


Figure 13. Flow chart for calibration of IDM.

Based on the optimized parameter set, the IDM results were calculated according to the test dataset, and the average results of the evaluation metrics (RMSE, MAE, MASE, and R^2) for the follower vehicles in the test dataset were calculated as shown in Table 7.

Table 7. The average of evaluation metrics for the test dataset, according to the IDM.

Model Type	Follower Acceleration				Follower Speed				Follower Position			
	RMSE	MAE	MASE	R^2	RMSE	MAE	MASE	R^2	RMSE	MAE	MASE	R^2
IDM	0.3953	0.3048	6.4268	0.5421	0.9497	0.7497	13.9983	0.8587	7.0966	5.4225	4.4596	0.9952

5. Experimental Results and Discussion

The ANN, LSTM, and Transformer models were applied according to the types of prediction methods mentioned in Section 3.1. The average results of the evaluation metrics (RMSE, MAE, MASE, and R^2) for the test dataset of follower vehicles are shown in Tables 8–10 for the ANN, LSTM, and Transformer models, respectively.

Table 8. Average of the evaluation metric results for the test dataset according to the type of ANN model.

Model Type	Follower Acceleration				Follower Speed				Follower Position			
	RMSE	MAE	MASE	R^2	RMSE	MAE	MASE	R^2	RMSE	MAE	MASE	R^2
ANN	0.3736	0.2981	6.2438	0.5632	1.0678	0.8301	15.628	0.8591	8.1775	6.3999	5.2565	0.9956
ANN & IPE	**0.3717**	**0.2956**	**6.1874**	**0.5664**	**1.0341**	**0.8062**	**15.158**	**0.8685**	**8.0029**	**6.2619**	**5.1372**	**0.9957**
Percentage of Improvement	0.49%	0.81%	0.9%	0.57%	3.16%	2.88%	3.01%	1.09%	2.13%	2.15%	2.27%	0.01%

Note: Bold numbers indicate the best values.

Table 9. Average of the evaluation metric results for the test dataset according to the type of LSTM model.

Model Type	Follower Acceleration				Follower Speed				Follower Position			
	RMSE	MAE	MASE	R^2	RMSE	MAE	MASE	R^2	RMSE	MAE	MASE	R^2
LSTM	0.3732	0.2979	6.2012	0.5682	1.0861	0.8479	15.888	0.8519	7.6267	5.8705	4.8332	0.9962
LSTM & IPE	**0.3712**	**0.2954**	**6.1595**	**0.5707**	**0.9537**	**0.7443**	**14.098**	**0.8792**	**6.5993**	**4.9967**	**4.1151**	**0.9968**
Percentage of Improvement	0.52%	0.85%	0.67%	0.44%	12.2%	12.2%	11.3%	3.21%	13.5%	14.9%	14.9%	0.06%

Note: Bold numbers indicate the best values.

Table 10. Average of the evaluation metric results for the test dataset according to the type of Transformer model.

Model Type	Follower Acceleration				Follower Speed				Follower Position			
	RMSE	MAE	MASE	R^2	RMSE	MAE	MASE	R^2	RMSE	MAE	MASE	R^2
Transformer	0.3686	0.2891	6.0267	0.5661	1.0311	0.8110	14.446	0.8241	10.016	7.5448	6.2142	0.9717
Transformer & IPE	**0.3611**	**0.2850**	**5.9529**	**0.5809**	**0.8692**	**0.6718**	**12.676**	**0.9009**	**6.0891**	**4.6686**	**3.8571**	**0.9972**
Percentage of Improvement	2.04%	1.42%	1.22%	2.62%	15.7%	17.2%	12.3%	9.32%	39.2%	38.1%	37.9%	2.62%

Note: Bold numbers indicate the best values.

As shown in Tables 8–10, the percentage of improvement for each evaluation metric (RMSE, MAE, MASE, and R^2) was calculated. The results indicate that all models using the proposed prediction method [Δv, Δx, Vn, IPE], which includes the variable of IPE analysis results, outperformed those using the traditional prediction method [Δv, Δx, Vn]. The percentage of improvements varied among the models, with the Transformer & IPE model showing the best performance. For the follower acceleration output, for example,

the percentage improvement was 2.04%, 1.42%, 1.22%, and 2.62% for RMSE, MAE, MASE, and R^2, in that order. The ANN & IPE model performed less well, with percentages of 0.49%, 0.81%, 0.9%, and 0.57%, respectively. Similar results were observed for the follower speed and follower position outputs. These improvement percentages demonstrate that adding IPE analysis results as a new input variable to the AI-based car-following models improved overall prediction performance for the outputs of follower acceleration, follower speed, and follower position.

The IDM was also used to evaluate the competence of the AI-based car-following models that used the proposed prediction method. The IDM was prepared and implemented to train and test the same real-world traffic data used. The average results of evaluation metrics (RMSE, MAE, MASE, and R^2) for the test dataset of follower vehicles are shown in Tables 11–13 for the ANN & IPE, LSTM & IPE, and Transformer & IPE models, respectively, with IDM. The percentage of improvement for each evaluation metric (RMSE, MAE, MASE, and R^2) was also calculated. The results indicate that all AI-based models using the proposed prediction method [Δv, Δx, Vn, IPE], which includes the variable of IPE analysis results, outperformed the IDM for the follower acceleration variable across all evaluation metrics. However, only the Transformer & IPE model outperformed the IDM for follower acceleration, follower speed, and follower position variables across all evaluation metrics. The percentages of improvement varied between satisfactory and unsatisfactory for the ANN & IPE and LSTM & IPE models, as shown in the tables, indicating that Transformer models outperformed the ANN and LSTM models in prediction.

In addition to the evaluation metrics, further tests were conducted to provide a comprehensive evaluation about incorporating the IPE analysis results into the AI-based models. It included statistical testing, ablation experiments, and feature contribution analysis.

The statistical one-tailed *t*-test method was conducted on different model pairs across three response variables (follower acceleration, follower speed, and follower position) to assess whether the first model in each comparison achieved significant improvement in reducing the prediction errors using the null hypothesis $Ho: \mu_1 - \mu_2 > 0$, where μ_1 and μ_2 refer to the mean absolute errors of the first and second models, respectively.

Table 11. Average of the evaluation metric results for the test dataset according to the IDM and ANN&IPE model.

Model Type	Follower Acceleration				Follower Speed				Follower Position			
	RMSE	**MAE**	**MASE**	**R^2**	**RMSE**	**MAE**	**MASE**	**R^2**	**RMSE**	**MAE**	**MASE**	**R^2**
IDM	0.3953	0.3048	6.4268	0.5421	0.9497	0.7497	13.998	0.8587	7.0966	5.4225	4.4596	0.9952
ANN & IPE	**0.3717**	**0.2956**	**6.1874**	**0.5664**	**1.0341**	**0.8062**	**15.158**	**0.8685**	**8.0029**	**6.2619**	**5.1372**	**0.9957**
Percentage of improvement	5.96%	3.01%	3.72%	4.49%	−8.9%	−7.5%	−8.3%	1.15%	−12%	−15%	−15%	0.05%

Note: Bold numbers indicate the best values.

Table 12. Average of the evaluation metric results for the test dataset according to the IDM and LSTM & IPE model.

Model Type	Follower Acceleration				Follower Speed				Follower Position			
	RMSE	**MAE**	**MASE**	**R^2**	**RMSE**	**MAE**	**MASE**	**R^2**	**RMSE**	**MAE**	**MASE**	**R^2**
IDM	0.3953	0.3048	6.4268	0.5421	0.9497	0.7497	13.998	0.8587	7.0966	5.4225	4.4596	0.9952
LSTM & IPE	**0.3712**	**0.2954**	**6.1595**	**0.5707**	**0.9537**	**0.7443**	**14.098**	**0.8792**	**6.5993**	**4.9967**	**4.1151**	**0.9968**
Percentage of Improvement	6.09%	3.1%	4.16%	5.28%	−0.4%	0.71%	−0.7%	2.4%	7%	7.85%	7.735	0.165

Note: Bold numbers indicate the best values.

Table 13. Average of the evaluation metric results for the test dataset according to the IDM and Transformer & IPE model.

Model Type	Follower Acceleration				Follower Speed				Follower Position			
	RMSE	MAE	MASE	R^2	RMSE	MAE	MASE	R^2	RMSE	MAE	MASE	R^2
IDM	0.3953	0.3048	6.4268	0.5421	0.9497	0.7497	13.998	0.8587	7.0966	5.4225	4.4596	0.9952
Transformer & IPE	**0.3611**	**0.2850**	**5.9529**	**0.5809**	**0.8692**	**0.6718**	**12.676**	**0.9009**	**6.0891**	**4.6686**	**3.8571**	**0.9972**
Percentage of Improvement	8.64%	6.51%	7.37%	7.15%	8.47%	10.4%	9.44%	4.92%	14.2%	13.9%	13.5%	0.2%

Note: Bold numbers indicate the best values.

For the comparisons (ANN & IPE–ANN, LSTM & IPE–LSTM, and Transformer & IPE–Transformer), the results revealed that all t-statistics were negative and associated with extremely small *p*-values far below the conventional significant level ($\alpha = 0.05$) shown in Table 14, providing strong evidence to reject the null hypothesis. This evidence confirms that by adding the IPE analysis results, the AI-based models consistently achieved lower mean absolute errors across all outputs (follower acceleration, follower speed, and follower position) than the models without the IPE analysis results.

Table 14. T-test comparisons for different models.

Model Type	Follower Acceleration		Follower Speed		Follower Position	
	t Stat	*p*-Value	t Stat	*p*-Value	t Stat	*p*-Value
ANN & IPE + ANN	−4.62	1.93×10^{-6}	−16.72	1.82×10^{-62}	−13.69	1.08×10^{-42}
LSTM & IPE + LSTM	−4.93	3.97×10^{-7}	−40.36	0	−52.97	0
Transformer & IPE + Transformer	−6.27	1.79×10^{-10}	−16.00	2.21×10^{-57}	−15.23	3.16×10^{-52}
ANN & IPE + IDM	−4.24	1.08×10^{-5}	15.68	3.58×10^{-55}	33.10	3.23×10^{-230}
LSTM & IPE + IDM	−5.75	4.39×10^{-9}	4.34	7.02×10^{-6}	−1.91	0.028
Transformer & IPE + IDM	−13.97	2.42×10^{-44}	−11.43	2.05×10^{-30}	−8.16	1.79×10^{-16}

In comparisons involving the IDM, the results revealed that only the Transformer & IPE model outperformed the IDM for all outputs, whereas the results of the ANN & IPE and LSTM & IPE models varied between satisfactory and unsatisfactory, as shown in Table 14. These results align with the results of the evaluation metrics comparing the IDM with the AI models shown in Tables 11–13, indicating a disparity in the predictive efficiency of the AI models. Transformer models demonstrated higher efficiency, while ANN models were less efficient, a difference attributed to the models' structure and development, particularly in their ability to handle complex data patterns and learn from larger datasets effectively.

In addition to the proposed prediction method, we used the leader position data ($X\iota$) as a new input variable, replacing the IPE analysis results of the leader position data to allow for further comparison. By systematically removing or substituting variables, the ablation study was conducted for the Transformer & IPE model to reveal the relative importance of each variable in prediction accuracy.

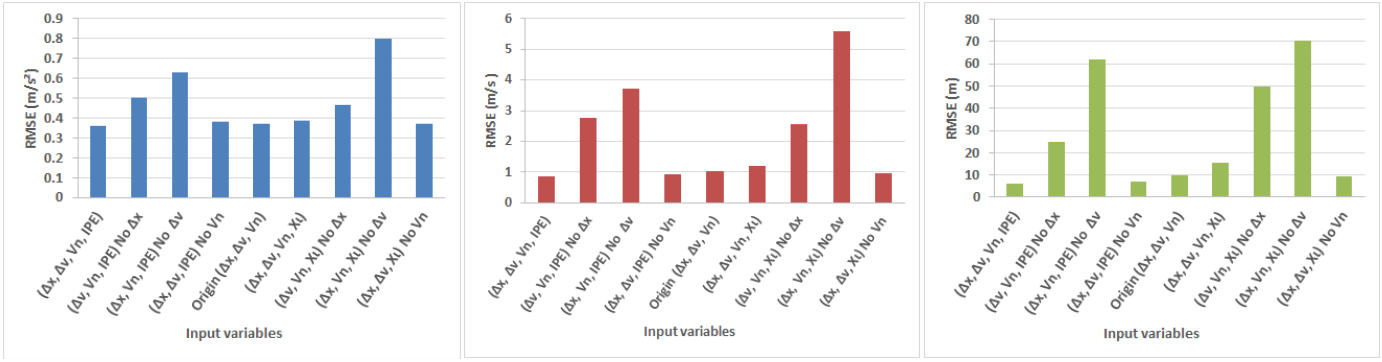
The results shown in Table 15 and Figure 14 reveal that the proposed prediction method (Δx, Δv, Vn, IPE), including IPE analysis results, produced the lowest RMSE values (0.3611, 0.8692, and 6.089) across all the outputs of follower acceleration, follower speed, and follower position, respectively. While replacing the IPE analysis results with leader position data $X\iota$ (Δx, Δv, Vn, $X\iota$) introduced worse performance across all the output, indicating that this variable does not enhance predictive accuracy, this result is evidence of the advantage of using IPE in analyzing leader position data to improve the prediction

accuracy of the AI-based car-following models instead of using leader position data directly. However, even if the individual contribution of IPE analysis results seems relatively small, its inclusion led to measurable performance gains.

Table 15. Ablation experiment comparison for Transformer models.

Input Variable Scenario	Follower Acceleration RMSE	Follower Speed RMSE	Follower Position RMSE
(Δx, Δv, Vn, IPE)	**0.3611**	**0.8692**	**6.0890**
(Δv, Vn, IPE) No Δx	0.5001	2.7549	24.9696
(Δx, Vn, IPE) No Δv	0.6297	3.7290	61.8244
(Δx, Δv, IPE) No Vn	0.3799	0.9296	6.9903
Origin (Δx, Δv, Vn)	0.3686	1.0311	10.0169
(Δx, Δv, Vn, Xι *)	0.3887	1.2129	15.3703
(Δv, Vn, Xι) No Δx	0.4649	2.5667	49.5486
(Δx, Vn, Xι) No Δv	0.7989	5.5709	70.2001
(Δx, Δv, Xι) No Vn	0.3705	0.9642	9.2909

* Xι: leader position, Note: Bold numbers indicate the best values.



(**a**) RMSE for follower acceleration output (**b**) RMSE for follower speed output (**c**) RMSE for follower position output

Figure 14. Ablation experiment comparison for Transformer model based on different input variables.

For the feature contribution analysis, four complementary importance techniques were used: perturbation analysis, gradient-based attribution, SHAP values, and permutation importance to quantify the contribution degree of each input variable in terms of the Transformer & IPE model.

The results shown in Table 16 reveal that IPE analysis results had a minimal contribution degree across perturbation analysis, gradient-based attribution, SHAP values, and permutation importance. However, this limitation can be attributed to the nature of variable interaction effects, where a variable with low marginal weight may still improve model performance when combined with other inputs. Thus, instead of serving as a primary driver, the collective evidence refers to IPE analysis results acting as a supportive variable that enhanced the capability of the model.

Table 16. Feature contribution analysis for Transformer models.

Variable Contribution	Perturbation Analysis	Gradient Based Attribution	SHAP Values	Permutation Importance
Δx	26.66	32.24	34.75	31.26
Δv	33.19	1.66	6.85	52.67

Table 16. *Cont.*

Variable Contribution	Perturbation Analysis	Gradient Based Attribution	SHAP Values	Permutation Importance
Vn	31.85	60.64	57.7	43.83
IPE	8.3	5.46	0.7	1.02

As indicated in the conclusion regarding the hypotheses in Section 2.1, the IPE analysis results have proven their ability to improve AI-based car-following models that predict the movement of follower vehicles, unlike real vehicle position data.

6. Conclusions

One of the advantages of IPE is its ease of direct application to local position data and interpretation of the results. It is also capable of converting vehicle position data from an unlimited range to a limited range between 0 and 0.3317. This study has identified that an IPE analysis result equal to 0.3317 indicates a streamlined movement, while any deviation from this value indicates a relative change in velocity. In other words, an IPE analysis result of 0.3317, based on the previously mentioned parameters, serves as the reference value for any vehicle traveling at a constant speed. Therefore, any slight variation in the IPE analysis results of lead vehicles will directly affect their follower vehicles, according to car-following theory.

This study adopted two types of prediction methods to examine the efficiency of IPE analysis results in developing artificial intelligence-based car-following models. First, we used the traditional prediction method, which includes relative velocity, relative distance, and velocity of the follower vehicle as input variables to predict the follower acceleration. Second, in addition to the variables in the first prediction method, we added the IPE analysis results of the lead vehicle's position data as a new input variable to predict the follower acceleration.

Three types of neural networks were adopted according to the development of artificial intelligence models: artificial neural networks (ANNs), long short-term memory networks (LSTMs), and Transformer models, and they were applied according to the types of prediction methods. The IDM was also used to evaluate the competence of the AI-based car-following models that used the proposed prediction method.

Four types of evaluation metrics (RMSE, MAE, MASE, and R^2), in addition to *t*-tests, ablation experiments, and feature contribution analysis, were used to provide a comprehensive evaluation of incorporating IPE analysis results into the baseline AI-based models. Overall, the integration of the findings shows that adding IPE analysis results improved the performance of the original models, and the Transformer & IPE model outperformed the other models. Although the individual contribution of IPE analysis appears relatively small, its inclusion produced a statistically significant reduction in prediction errors.

The IPE is effective for analyzing real trajectory data, and its results accurately reflect vehicle behavior as measured by car-following models, thereby supporting the recommendation to use it in analysis of car-following theory.

For the proposed approach, the experimental analysis is based only on data from lane 1 of the NGSIM US-101 dataset, which represents a limitation. Therefore, future research should consider incorporating more diverse, more recent datasets. Furthermore, future research on using the IPE in shockwave analysis and comparisons among different types of entropy is suggested.

Finally, IPE enables researchers to improve existing car-following models and traffic flow theories, including trajectory planning, driver assistance devices, adaptive cruise

control (ACC), and cooperative adaptive cruise control (CACC). It will serve as a foundation for future studies utilizing these technologies and their applications in intelligent transportation systems and autonomous and connected vehicles.

Author Contributions: Conceptualization, A.M.S. and İ.Ş.; Methodology, A.M.S. and İ.Ş.; Software, A.M.S.; Validation, İ.Ş.; Formal analysis, A.M.S. and İ.Ş.; Investigation, A.M.S. and İ.Ş.; Resources, A.M.S. and İ.Ş.; Data curation, A.M.S. and İ.Ş.; Writing—original draft, A.M.S.; Writing—review & editing, İ.Ş.; Visualization, A.M.S.; Supervision, İ.Ş. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

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

Appendix A

To explain the IPE algorithm, let us consider a one-dimensional time series signal denoted as

$$x = \{x_i\}_{i=1}^{N} \tag{A1}$$

where N is the length of the time series.

Step 1: Partition the one-dimensional time series x into a matrix of overlapping column vectors.

This partitioning uses two hyperparameters:

D = The embedding dimension, which controls the length of each of the new column vectors.

τ = The embedding time delay, which controls the number of time periods between elements of each of the new column vectors.

Selecting the embedding dimension D and the time delay τ for the time series gives us the $D \times J$ reconstruction matrix, and the sample data in Equation (A1) are partitioned as follows:

$$X = \begin{bmatrix} x_{(1)} & x_{(2)} & \cdots & x_j \\ x_{(1+\tau)} & x_{(2+\tau)} & \cdots & x_{(j+\tau)} \\ \vdots & \vdots & \ddots & \vdots \\ x_{(1+(D-1)\tau} & x_{(2+(D-1)\tau} & \cdots & x_{(j+(D-1)\tau} \end{bmatrix} \tag{A2}$$

where j denotes the column number in the reconstruction matrix.

$J = N - (D-1)\tau$ denotes the total number of columns in the reconstruction matrix.

Step 2: Digitize the first row of the reconstruction matrix in Equation (A2) $X\ (1, j)$ by the uniform quantization $UQ(x_i)$ function and place it in the first row of the permutation matrix $S\ (1, j)$.

The uniform quantization $UQ(x_i)$ function (A3) converts each value of the first row of the reconstruction matrix into an integer digit l whose value is $0 \leq l \leq L-1$.

$$UQ(x_i) = l \qquad l = (0, 1, \ldots, L-1)$$

$$UQ(x_i) = \begin{cases} 0 & x_{min} + l\Delta \le x_i < x_{min} + (l+1)\Delta \\ 1 & x_{min} + l\Delta \le x_i < x_{min} + (l+1)\Delta \\ . & . \\ . & . \\ . & . \\ L-1 & x_{min} + l\Delta \le x_i \le x_{min} + (l+1)\Delta \end{cases} \tag{A3}$$

$$\Delta = \frac{(x_{max} - x_{min})}{L} \tag{A4}$$

where x_{min} and x_{max} represent the minimum and maximum values of the one-dimensional time series x, respectively.

L is the number of discretization levels (quantization levels).

Δ is the spacing between the quantization levels (A4).

Step 3: Afterwards, other rows of the permutation matrix $S\ (n, j)$ (where $2 \le n \le D$) are obtained from Equation (A5), based on the first row of reconstruction matrix (A2), $X\ (1, j)$.

$$S_{(n,j)} = S_{(1,j)} + \left\lfloor \frac{x_{(n,j)} - x_{(1,j)}}{\Delta} \right\rfloor \tag{A5}$$

$1 \le j \le N - (D-1)\tau$,

$2 \le n \le D$,

where $\lfloor\ \rfloor$ represents a function that rounds the elements in it to the nearest integers towards zero.

Finally, S is defined as the $D \times J$ permutation matrix. Each column of permutation matrix S will correspond to one of the possible permutation patterns π_l, $1 \le l \le L^D$, i.e., there are L^D patterns.

The possible permutation patterns π_l, ($1 \le l \le L^D$), will be equal to L^D patterns [45,48], i.e., when $D = L = 3$, there are $3^3 = 27$ patterns, as shown in Table A1 tbelow.

Table A1. The possible permutation patterns according to L^D.

$\pi_1 = \{0,0,0\}$	$\pi_{10} = \{1,0,0\}$	$\pi_{19} = \{2,0,0\}$
$\pi_2 = \{0,0,1\}$	$\pi_{11} = \{1,0,1\}$	$\pi_{20} = \{2,0,1\}$
$\pi_3 = \{0,0,2\}$	$\pi_{12} = \{1,0,2\}$	$\pi_{21} = \{2,0,2\}$
$\pi_4 = \{0,1,0\}$	$\pi_{13} = \{1,1,0\}$	$\pi_{22} = \{2,1,0\}$
$\pi_5 = \{0,1,1\}$	$\pi_{14} = \{1,1,1\}$	$\pi_{23} = \{2,1,1\}$
$\pi_6 = \{0,1,2\}$	$\pi_{15} = \{1,1,2\}$	$\pi_{24} = \{2,1,2\}$
$\pi_7 = \{0,2,0\}$	$\pi_{16} = \{1,2,0\}$	$\pi_{25} = \{2,2,0\}$
$\pi_8 = \{0,2,1\}$	$\pi_{17} = \{1,2,1\}$	$\pi_{26} = \{2,2,1\}$
$\pi_9 = \{0,2,2\}$	$\pi_{18} = \{1,2,2\}$	$\pi_{27} = \{2,2,2\}$

Step 4: Compute the probability distribution P_l of each pattern π_l, where P_l is the probability of pattern π_l, which is computed from the relative frequency occurrence of π_l in permutation matrix $[S]$. After that the IPE can be computed according to Equation (1) in Section 2.1. Figure A1 shows the example calculation for the first scenario, stopping without movement, presented in Section 2.1.1, and Figure A2 shows the example calculation for the second scenario, one type of movement (free flow or congestion) without stopping, presented in Section 2.1.2.

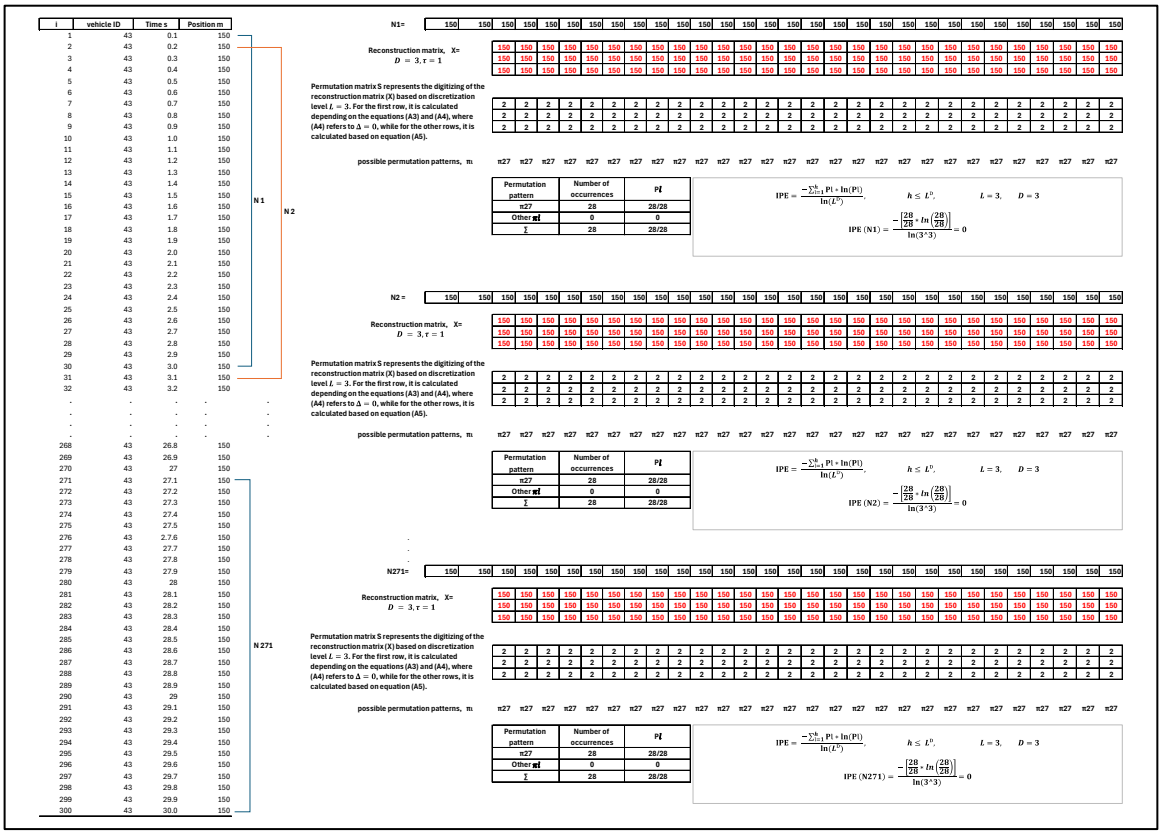


Figure A1. The IPE calculation method for the scenario of stopping without movement for vehicle ID 1.

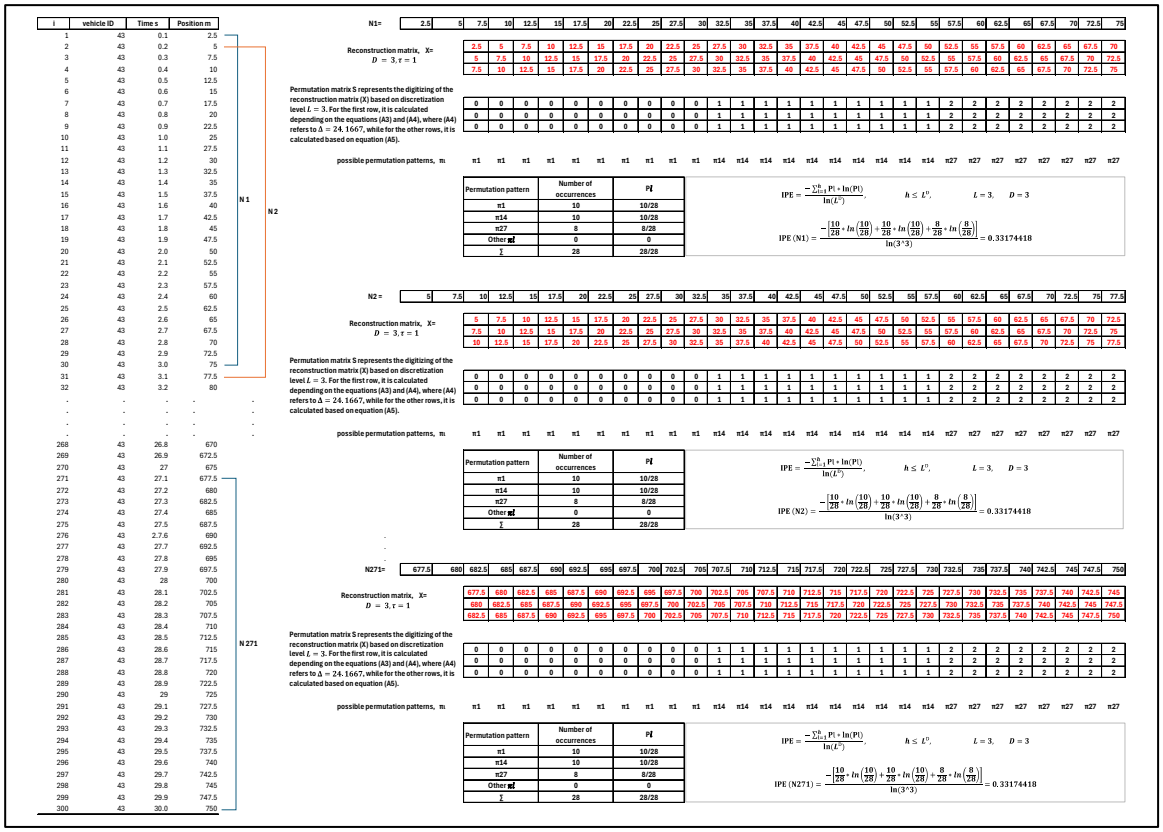


Figure A2. The IPE calculation method for the scenario of free flow or congested movement for vehicle ID 1.

References

1. Zhong, R.X.; Fu, K.Y.; Sumalee, A.; Ngoduy, D.; Lam, W.H.K. A cross-entropy method and probabilistic sensitivity analysis framework for calibrating microscopic traffic models. *Transp. Res. Part C Emerg. Technol.* **2016**, *63*, 147–169. [CrossRef]
2. Ahmed, H.U.; Huang, Y.; Lu, P. A review of car-following models and modeling tools for human and autonomous-ready driving behaviors in micro-simulation. *Smart Cities* **2021**, *4*, 314–335. [CrossRef]
3. Qin, P.; Li, X.; Bin, S.; Wu, F.; Pang, Y. Research on transformer and long short-term memory neural network car-following model considering data loss. *Math. Biosci. Eng.* **2023**, *20*, 19617–19635. [CrossRef]
4. Zhu, K.; Yang, X.; Zhang, Y.; Liang, M.; Wu, J. A Heterogeneity-Aware Car-Following Model: Based on the XGBoost Method. *Algorithms* **2024**, *17*, 68. [CrossRef]
5. Zhang, L. Swarm Intelligent Car-Following Model for Autonomous Vehicle Platoon Based on Particle Swarm Optimization Theory. *Electronics* **2025**, *14*, 1851. [CrossRef]
6. Cazis, D.C.; Herman, R.; Rothery, R.W. Nonlinear Follow-The-Leader Models of Traffic Flow. *Oper. Res.* **1961**, *9*, 545–567.
7. Newell, G.F. Nonlinear Effects in the Dynamics of Car Following. *Oper. Res.* **1961**, *9*, 209–229. [CrossRef]
8. Gipps, P.G. A Behavioural Car-Following Model for Computer Simulation. *Transp. Res. Part B Methodol.* **1981**, *15*, 105–111. [CrossRef]
9. Treiber, M.; Hennecke, A.; Helbing, D. Congested Traffic States in Empirical Observations and Microscopic Simulations. *arXiv* **2000**. [CrossRef] [PubMed]
10. Jiang, R.; Wu, Q.; Zhu, Z. Full velocity difference model for a car-following theory. *Phys. Rev. E* **2001**, *64*, 017101. [CrossRef]
11. Zhang, T.; Jin, P.J.; McQuade, S.T.; Bayen, A.; Piccoli, B. Car-Following Models: A Multidisciplinary Review. *IEEE Trans. Intell. Veh.* **2025**, *10*, 92–116. [CrossRef]
12. Kehtarnavaz, N.; Member, S.; Griswold, N.; Miller, K.; Lescoe, P. A Transportable Neural-Network Approach to Autonomous Vehicle Following. *IEEE Trans. Veh. Technol.* **1998**, *47*, 694–702. [CrossRef]
13. Panwai, S.; Dia, H. Neural agent car-following models. *IEEE Trans. Intell. Transp. Syst.* **2007**, *8*, 60–70. [CrossRef]
14. Khodayari, A.; Ghaffari, A.; Kazemi, R.; Braunstingl, R. A modified car-following model based on a neural network model of the human driver effects. *IEEE Trans. Syst. Man Cybern. Part A Syst. Hum.* **2012**, *42*, 1440–1449. [CrossRef]
15. Zhou, M.; Qu, X.; Li, X. A recurrent neural network based microscopic car following model to predict traffic oscillation. *Transp. Res. Part C Emerg. Technol.* **2017**, *84*, 245–264. [CrossRef]
16. Huang, X.; Sun, J.; Sun, J. A car-following model considering asymmetric driving behavior based on long short-term memory neural networks. *Transp. Res. Part C Emerg. Technol.* **2018**, *95*, 346–362. [CrossRef]
17. Wu, F.; Work, D.B. Connections between classical car following models and artificial neural networks. In *Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018*; IEEE: New York, NY, USA, 2018.
18. Ma, L.; Qu, S. A sequence to sequence learning based car-following model for multi-step predictions considering reaction delay. *Transp. Res. Part C Emerg. Technol.* **2020**, *120*, 102785. [CrossRef]
19. Colombaroni, C.; Fusco, G.; Isaenko, N. Modeling Car following with Feed-Forward and Long-Short Term Memory Neural Networks. *Transp. Res. Procedia* **2021**, *52*, 195–202. [CrossRef]
20. Zhou, J.; Wan, J.; Zhu, F. Transfer Learning Based Long Short-Term Memory Car-Following Model for Adaptive Cruise Control. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 21345–21359. [CrossRef]
21. Qu, D.; Wang, S.; Liu, H.; Meng, Y. A Car-Following Model Based on Trajectory Data for Connected and Automated Vehicles to Predict Trajectory of Human-Driven Vehicles. *Sustainability* **2022**, *14*, 7045. [CrossRef]
22. Qin, P.; Li, H.; Li, Z.; Guan, W.; He, Y. A CNN-LSTM Car-Following Model Considering Generalization Ability. *Sensors* **2023**, *23*, 660. [CrossRef]
23. Adewale, A.; Lee, C. Prediction of Car-Following Behavior of Autonomous Vehicle and Human-Driven Vehicle Based on Drivers' Memory and Cooperation with Lead Vehicle. *Transp. Res. Rec.* **2024**, *2678*, 248–266. [CrossRef]
24. Wang, K.; Liu, Y.; Zhang, J. Modeling Human-Like Car-Following Model for Intelligent Vehicles Based on Deep Reinforcement Learning. *Int. J. Automot. Technol.* **2025**, *27*, 163–172. [CrossRef]
25. Li, T.; Halatsis, A.; Stern, R. RACER: Rational Artificial Intelligence Car-Following-Model Enhanced by Reality. *IEEE Trans. Intell. Transp. Syst.* **2025**, *26*, 21199–21214. [CrossRef]
26. Liu, Y.; Wang, S.; Panicker, A.; Embry, K.; Asanova, A.; Li, T. A Phase-Aware AI Car-Following Model for Electric Vehicles with Adaptive Cruise Control: Development and Validation Using Real-World Data. *arXiv* **2025**. [CrossRef]
27. Petrov, A.I. Information and Entropy Aspects of the Specifics of Regional Road Traffic Accident Rate in Russia. *Information* **2023**, *14*, 138. [CrossRef]
28. Azami, H.; Sanei, S.; Rajji, T.K. Ensemble entropy: A low bias approach for data analysis. *Knowl. Based Syst.* **2022**, *256*, 109876. [CrossRef]

29. Zanin, M.; Zunino, L.; Rosso, O.A.; Papo, D. Permutation entropy and its main biomedical and econophysics applications: A review. *Entropy* **2012**, *14*, 1553–1577. [CrossRef]
30. Liu, Z.; Xu, C.; Chen, L.; Zhou, S. Dynamic Traffic Flow Entropy Calculation Based on Vehicle Spacing. In *IOP Conference Series: Earth and Environmental Science*; Institute of Physics Publishing: Bristol, UK, 2019. [CrossRef]
31. Lyu, W.; Gonçalves, R.; Guo, F.; Torrão, G.; Radhakrishnan, V.; Guillen, P.P.; Louw, T.; Merat, N. Applying Entropy to Understand Drivers' Uncertainty during Car-following. In Proceedings of the 12th International Conference on Measuring Behavior and the 6th Seminar on Behavioral Methods, Krakow, Poland, 13–15 October 2021; pp. 13–15. [CrossRef]
32. Zhang, Z.; Xiang, Z.; Chen, Y.; Xu, J. Fuzzy permutation entropy derived from a novel distance between segments of time series. *AIMS Math.* **2020**, *5*, 6244–6260. [CrossRef]
33. Liu, Z.; Wang, Y.; Cheng, Q.; Yang, H. Analysis of the Information Entropy on Traffic Flows. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 18012–18023. [CrossRef]
34. Tang, L. Application of information entropy algorithm in safety risk prediction of road traffic driving behaviors. *Adv. Transp. Stud.* **2023**, *1*, 147–158. [CrossRef]
35. Baldini, G. On the application of entropy measures with sliding window for intrusion detection in automotive in-vehicle networks. *Entropy* **2020**, *22*, 1044. [CrossRef] [PubMed]
36. Guo, C.; Zhang, J.; Cao, Z. Traffic congestion recognition based on information entropy. In *Proceedings of the International Conference on Smart Transportation and City Engineering*; SPIE: Bellingham, WA, USA, 2021; pp. 523–528. [CrossRef]
37. Cui, Z.; Chen, G.; Liu, B.; Li, D. A Multiscale Symbolic Dynamic Entropy Analysis of Trac Flow. *J. Adv. Transp.* **2022**, *2022*, 8389229. [CrossRef]
38. Ye, W.; Xu, Y.; Shi, X.; Shiwakoti, N.; Ye, Z.; Zheng, Y. A macroscopic safety indicator for road segment: Application of entropy theory. *Phys. A Stat. Mech. Its Appl.* **2024**, *642*, 129787. [CrossRef]
39. Zhou, J.; Li, Y.; Wang, M. Research on the Threshold Determination Method of the Duffing Chaotic System Based on Improved Permutation Entropy and Poincaré Mapping. *Entropy* **2023**, *25*, 1654. [CrossRef] [PubMed]
40. Zhou, J.; Hao, B.; Li, Y.; Yang, X. Underwater Small Target Detection Method Based on the Short-Time Fourier Transform and the Improved Permutation Entropy. *Acoustics* **2024**, *6*, 870–884. [CrossRef]
41. Zheng, K.; Gan, H.S.; Chaw, J.K.; Teh, S.H.; Chen, Z. Generalized Gaussian Distribution Improved Permutation Entropy: A New Measure for Complex Time Series Analysis. *Entropy* **2024**, *26*, 960. [CrossRef]
42. Bandt, C.; Pompe, B. Permutation Entropy: A Natural Complexity Measure for Time Series. *Phys. Rev. Lett.* **2002**, *88*, 174102. [CrossRef]
43. Duran, O.; Fumagalli, L.; Arata, A. Permutation Entropy and Ordinal Patterns as a Resilience Indicator. In *Proceedings of the 2023 7th International Conference on System Reliability and Safety, ICSRS 2023*; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2023; pp. 384–388. [CrossRef]
44. Cuesta-Frau, D. Permutation entropy: Influence of amplitude information on time series classification performance. *Math. Biosci. Eng.* **2019**, *16*, 6842–6857. [CrossRef]
45. Chen, Z.; Li, Y.; Liang, H.; Yu, J. Improved permutation entropy for measuring complexity of time series under noisy condition. *Complexity* **2019**, *2019*, 1403829. [CrossRef]
46. Shahatha, A.M.; Şahin, İ. Permutation Entropy and Improved Permutation Entropy Applied to Traffic Flow Variables. In *Graduate Student Conference—Transportation—2025 Fall Semester 22/11/2025*; TMMOB Chamber of Civil Engineers Istanbul Branch: Istanbul, Turkey, 2025. Available online: https://ogrenci.imo.org.tr/index-eng.html#kitap (accessed on 14 December 2025).
47. Zunino, L.; Olivares, F.; Scholkmann, F.; Rosso, O.A. Permutation entropy based time series analysis: Equalities in the input signal can lead to false conclusions. *Phys. Lett. Sect. A Gen. At. Solid State Phys.* **2017**, *381*, 1883–1892. [CrossRef]
48. Mansourian, N.; Sarafan, S.; Ghirmai, T.; Cao, H.; Azar, F.T. Novel QRS Detection Based on the Adaptive Improved Permutation Entropy. *Biomed. Signal Process. Control* **2022**, *80*, 104270. [CrossRef]
49. Blvd, V. Vehicle Trajectory Study Area On-Ramp Cahuenga Blvd. Off-Ramp. 2007. Available online: https://ops.fhwa.dot.gov/trafficanalysistools/ngsim.htm (accessed on 23 February 2026).
50. Moller, M.F. A scaled conjugate gradient algorithm for fast supervised learning. *Neural Netw.* **1993**, *6*, 525–533. [CrossRef]
51. Haykin, S.S. *Neural Networks and Learning Machines*, 3rd ed.; Prentice Hall/Pearson: Hamilton, ON, Canada, 2009.
52. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
53. Tepetidis, N.; Koutsoyiannis, D.; Iliopoulou, T.; Dimitriadis, P. Investigating the Performance of the Informer Model for Streamflow Forecasting. *Water* **2024**, *16*, 2882. [CrossRef]
54. Krichen, M.; Mihoub, A. Long Short-Term Memory Networks: A Comprehensive Survey. *AI* **2025**, *6*, 215. [CrossRef]
55. Lindemann, B.; Maschler, B.; Sahlab, N.; Weyrich, M. A Survey on Anomaly Detection for Technical Systems using LSTM Networks. *Comput. Ind.* **2024**, *131*, 103498. [CrossRef]
56. Hua, Y.; Zhao, Z.; Li, R.; Chen, X.; Liu, Z.; Zhang, H. Deep Learning with Long Short-Term Memory for Time Series Prediction. *arXiv* **2018**. [CrossRef]

57. Understanding LSTM Networks—Colah's Blog. Available online: https://colah.github.io/posts/2015-08-Understanding-LSTMs/ (accessed on 6 March 2026).
58. Zhao, J.; Li, X.; Xue, Q.; Zhang, W. Spatial-Channel Transformer Network for Trajectory Prediction on the Traffic Scenes. *arXiv* **2021**. [CrossRef]
59. Pazho, A.D.; Noghre, G.A.; Katariya, V.; Tabkhi, H. VT-Former: An Exploratory Study on Vehicle Trajectory Prediction for Highway Surveillance Through Graph Isomorphism and Transformer. *arXiv* **2024**. [CrossRef]
60. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. *arXiv* **2023**. [CrossRef]
61. What Is a Transformer Model? | IBM. Available online: https://www.ibm.com/think/topics/transformer-model#1280257394 (accessed on 23 February 2026).
62. Dong, S.; Wang, P.; Abbas, K. *A Survey on Deep Learning and Its Applications*; Elsevier Ireland Ltd.: Amsterdam, The Netherlands, 2021. [CrossRef]
63. Srinivas, A.; Lin, T.Y.; Parmar, N.; Shlens, J.; Abbeel, P.; Vaswani, A. Bottleneck transformers for visual recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*; IEEE Computer Society: New York, NY, USA, 2021; pp. 16514–16524. [CrossRef]
64. Hung, W.-C.; Kretzschmar, H.; Lin, T.-Y.; Chai, Y.; Yu, R.; Yang, M.-H.; Anguelov, D. SoDA: Multi-Object Tracking with Soft Data Association. *arXiv* **2020**. [CrossRef]
65. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An Image Is Worth 16 × 16 Words: Transformers for Image Recognition at Scale. *arXiv* **2020**. [CrossRef]
66. Khan, S.; Naseer, M.; Hayat, M.; Zamir, S.W.; Khan, F.S.; Shah, M. Transformers in Vision: A Survey. *ACM Comput. Surv.* **2022**, *54*, 1–41. [CrossRef]
67. Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; Zhang, W. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. *arXiv* **2020**. [CrossRef]
68. Zhou, S.; Zheng, S.; Tian, J.; Jiang, R.; Zhang, H.M. Twenty-Five Years of the Intelligent Driver Model: Foundations, Extensions, Applications, and Future Directions. *arXiv* **2025**. [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.