data

MDPI

Article

ADAS Simulation Result Dataset Processing Based on Improved BP Neural Network

Songyan Zhao [1], Lingshan Chen [1,*] and Yongchao Huang [2]

[1] College of Mechanical and Automotive Engineering, Shanghai University of Engineering Science, Shanghai 200335, China; m310121401@sues.edu.cn
[2] Shanghai Waylancer Automobile Technology Co., Ltd., Shanghai 201805, China
* Correspondence: 17368152685@163.com; Tel.: +86-173-6815-2685

Abstract: The autonomous driving simulation field lacks evaluation and forecasting systems for simulation results. The data obtained from the simulation of target algorithms and vehicle models cannot be reasonably estimated. This problem affects subsequent vehicle improvement and parameter calibration. The authors relied on the simulation results of the AEB algorithm. We selected the BP Neural Network as the basis and improved it with a genetic algorithm optimized via a roulette algorithm. The regression evaluation indicators of the prediction results show that the GA-BP neural network has better prediction accuracy and generalization ability than the original BP neural network and other optimized BP neural networks. This GA-BP neural network also fills the Gap in Evaluation and Prediction Systems.

Keywords: ADAS functionality; simulation testing; BP neural networks; genetic algorithm

Citation: Zhao, S.; Chen, L.; Huang, Y. ADAS Simulation Result Dataset Processing Based on Improved BP Neural Network. *Data* **2024**, *9*, 11. https://doi.org/10.3390/data9010011

Academic Editor: Florentino Fdez-Riverola

Received: 14 October 2023
Revised: 22 November 2023
Accepted: 2 January 2024
Published: 5 January 2024

1. Introduction

ADAS is the abbreviation of Advanced Driver Assistance System. In brief, ADAS is designed to make active judgments and preventive measures before the driver's subjective reaction in an emergency [1].

ADAS uses sensors such as cameras, radar, lasers, and ultrasound to detect light, heat, pressure, and other parameters. These sensors are usually mounted in the front and rear bumpers, side mirrors, inside the driving lever, or windshield. Early ADAS technologies were mainly based on passive alarms. This system could alert drivers to abnormal conditions when a vehicle detects a potential hazard. Today's ADAS is capable of active intervention and emergency avoidance [2].

While automobiles can alter people's habits and lifestyles, they also create safety hazards and risk of accidents. It is noteworthy that traffic accidents due to untimely braking occur frequently. These accidents have resulted in an increasing number of deaths and huge economic losses.

Automatic Emergency Braking (AEB) could prevent or minimize damage caused by rear-end collisions, safeguarding the driver and passengers. This feature has become a fundamental Advanced Driver Assistance Systems (ADAS) aspect. In order to ensure that the AEB system triggers and operates reliably in real-life scenarios, the study of its testing techniques is essential. Researchers should establish a comprehensive operational plan and conduct extensive testing in various scenarios to analyze possible causes of AEB failure. These studies and measures can improve the active safety performance of AEB in several dimensions. From another perspective, we can also improve the stability and robustness of algorithms in these studies [3].

The testing of autonomous driving algorithms is typically divided into two categories: accurate vehicle testing and virtual simulation testing. Virtual simulation testing uses software to simulate real-world environments and traffic conditions. The vehicle is controlled

by the algorithm under test in the simulation software. Simulation testing has several advantages that help overcome the limitations of accurate vehicle testing. Simulation testing has unparalleled benefits that help overcome the limitations of accurate vehicle testing.

Nevertheless, the current simulation testing process needs a comprehensive evaluation framework and procedure to assess the outcomes after completing algorithm simulations. In addition, due to the lack of structure and coherence of the simulation data of ADAS functions [4], parameters and indicators specified in regulations and standards are difficult to apply to simulation test data evaluation. Furthermore, regulatory standards fail to establish a clear link between the parameters derived from simulations and the final evaluation results. As a result, an efficient evaluation framework is necessary to ascertain whether the algorithm meets the required criteria.

2. Methods

AEB Algorithm Simulation Test Process

The software utilized in this parking simulation is PreScan 8.5.0, renowned for its wide adoption in the field of autonomous driving simulation. PreScan integrates several functions, including scene establishment, sensor simulation models, basic vehicle dynamics models, algorithm interfaces, and interfaces with third-party software like CarSim and CARLA [5–7].

In this study, the co-simulation platform of PreScan and Simulink is employed, leveraging PreScan's built-in vehicle exterior and dynamics models. The first step involves constructing a straight AEB standard regulation scene, with a road length of 300 m and five lanes, each with a width of 3.5 m. The surrounding environment is simulated to include ordinary turf and cedar trees [8]. For the vehicle model, a black Audi A8 Sedan is selected, utilizing PreScan's own dynamic model and vehicle exterior dimensions.

The AEB trigger target vehicle is represented by PreScan's inflatable vehicle model, positioned at fixed distances of 100 m, 150 m, 200 m, and 250 m from the main vehicle's initial position. Each of these positions undergoes 50 experiments, resulting in a comprehensive evaluation [9]. To enable perception of the surrounding environment and detection of the target vehicle, the main vehicle is equipped with two millimeter-wave radars mounted on its bumper—a long-range radar and a short-range radar [10]. Additionally, the vehicle is equipped with a forward-facing camera positioned on the windshield and a LIDAR sensor mounted on the roof. These sensors play a vital role in the autonomous emergency braking (AEB) system by accurately perceiving the environment and detecting the target vehicle.

Throughout the simulations, PreScan's sensor models precisely simulate the behavior and detection capabilities of the millimeter-wave radars, camera, and LIDAR (Table 1). These models provide crucial inputs to the AEB algorithm, which determines the optimal timing for initiating braking based on the detected distance and relative velocity of the target vehicle. By utilizing the co-simulation platform of PreScan and Simulink, the AEB algorithm can be developed and tested within Simulink, utilizing the sensor inputs from PreScan. This integration allows for a comprehensive evaluation of the AEB system's performance across diverse scenarios, offering valuable insights for the development of autonomous driving technologies (Figure 1).

Table 1. Vehicle Radar Parameters.

	Long-Range Radar Tis1 Parameters	**Short Range Radar Tis2 Parameters**
FoV shape	pyramid type	pyramid type
Number of beams	Single beam	Single beam
Scanning frequency	25 Hz	25 Hz
Detection range	160 m	30 m
Transverse FOV opening angle	9 deg	75 deg
Number of detected targets	32	32

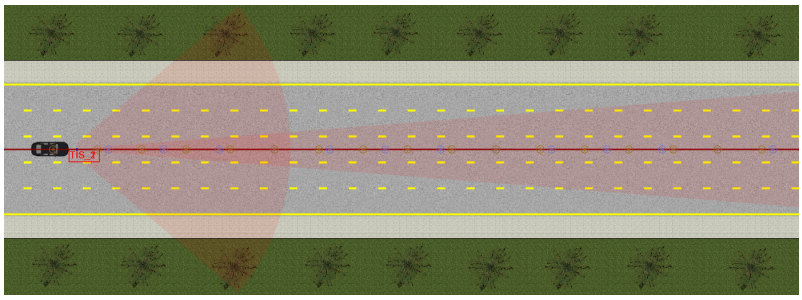

Figure 1. The distance schematic of Prescan main vehicle detection.

After setting the parameters and clicking build, Prescan will automatically compile the .slx file that can be opened in Simulink.

Then, we open Matlab in the process manager that comes with Prescan, and open the .slx file in the left directory. It contains the algorithm for controlling the main vehicle, dynamics model, and data interface. The Simulink interface after importing the algorithm is as shown in Figure 2.

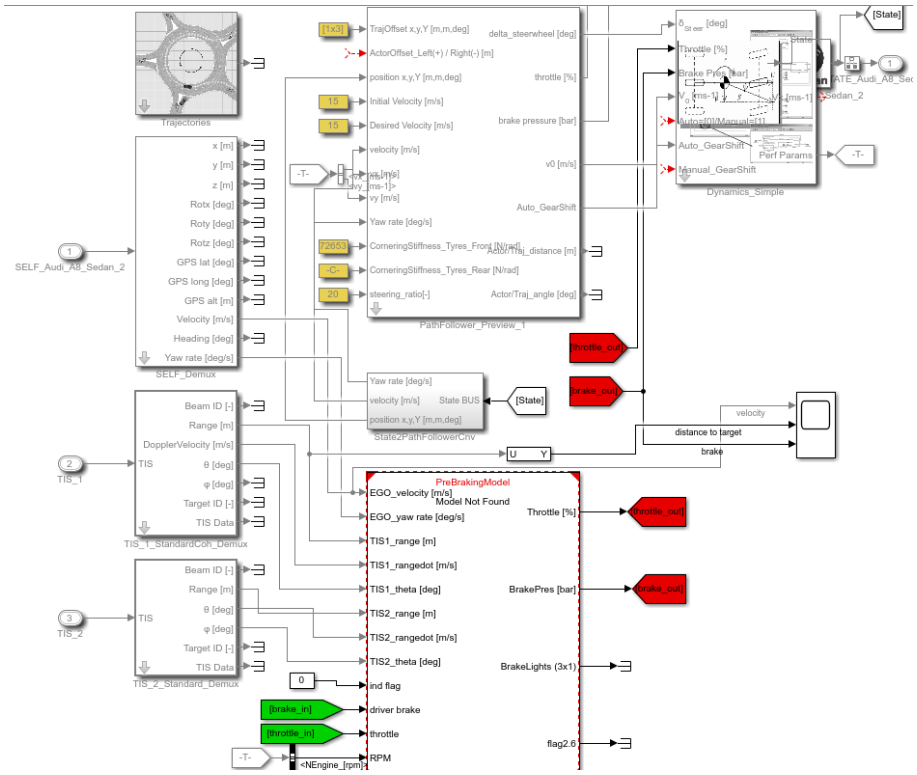

Figure 2. Simulink interface after importing the AEB algorithm.

In Figure 2, the lower module represents the AEB control algorithm area. Positioned at the bottom left, TIS1 and TIS2 modules correspond to the main vehicle's millimeter-wave radar modules, as showcased in Figure 3, which illustrates their detection range during the experiment. The pink line area (SRR) in Figure 3 shows the detection range of the Short Range Radar Tis2. The blue line area (LRR) in Figure 3 shows the detection range of the Long Range Radar Tis1.

On the left side, we find the GPS positioning and speed control module for the vehicle, ensuring accurate positioning and regulating the vehicle's speed accordingly. On the right side, the corresponding vehicle dynamics model provided by PreScan is displayed, enabling accurate simulation of the vehicle's dynamic behavior.

Once the rest of the data interfaces are connected, the simulation program and the 3D visualization interface can be executed [11]. This integration allows for seamless data

transfer between the simulation program and the 3D visualization interface, providing a comprehensive and immersive visual representation of the simulated scenario.

After completing a single simulation, the three data outputs include the main vehicle speed (v), distance from the target vehicle (d), and braking effort (B), which are then imported into the oscilloscope [12].

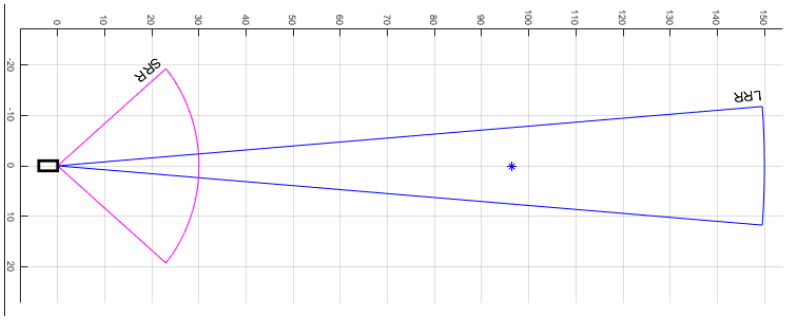

Figure 3. Main vehicle millimeter−wave radar detection range. * Detected objects.

3. Related Work

3.1. Baseline Neural Networks

The choice of baseline neural networks is huge, including BP, RBF, GRNN, PNN, and other neural networks. Among these networks, RBF is more similar to BP. They could approximate any nonlinear function with arbitrary accuracy. However, because they use different activation functions (RBF can only use radial basis function, but BP neural networks can use multiple activation functions), the performance and speed of their approximations are different. In general, RBF neural networks are more suitable for approximating continuous functions [13]. It is a forward neural network. Furthermore, compared with the BP neural network, the weights of the connections between the hidden layer and the input layer are not randomly determined but are fixed. Therefore, the application scope of the RBF neural network is not as comprehensive as the BP neural network. For subsequent algorithmic improvement, we chose the BP neural network as the baseline neural network.

The BP neural network is a typical nonlinear algorithm. BP neural networks consist of input layers, output layers, and several layers (one or more) hidden layers in between, each of which can have several nodes. The connection status of nodes between layers is represented by weights.

Forward propagation refers to the process of transmitting data or signals through a neural network. It involves passing the input through the network's layers, multiplying it by the corresponding weights, and then summing the results. The computed output is passed through an activation function, and the resulting computation is used as input for the next node in the network. This sequential computation continues until the final output is obtained. For example, in a perceptron, the input x is sequentially processed through the layers to obtain the output. This entire process is known as the forward propagation phase [14].

On the other hand, backpropagation is the process of comparing the network's output with the desired output. It involves calculating the error between the obtained output and the ideal output and then using this error to update the network's weights. Backpropagation is essentially a feedback process, where the error is propagated back through the network to adjust the weights. This iterative process involves continuously adjusting the weights among the network nodes to minimize the error over multiple iterations. Backpropagation is a crucial step in training neural networks to optimize their performance.

3.1.1. Building BP Neural Networks

Each layer of a neural network performs a linear summation process between its inputs and weights. However, purely linear transformations are insufficient to solve complex problems. To introduce nonlinearity into the network, an activation function is

used. The activation function is applied to the linear combination of inputs in each neuron, allowing the neural network to approximate any other nonlinear function. This enables the network to learn and make predictions based on the given simulation data.

The choice of activation function depends on the specific requirements of the model. In this paper, the tansig function is selected as the activation function for the model:

$$tansig(x) = \frac{2}{1 + e^{-2x}} - 1 \tag{1}$$

BP neural networks' commonly used activation functions are tansig, logsig, and others. The two most widely used are tansig and logsig; the only difference between them is that tansig performs a translation operation on top of logsig.

The choice of activation function depends on the specific requirements of the model. In this paper, because tansig maps the input to a range of $(-1, 1)$ and has an S-shaped curve, it is selected as the activation function for the model. The active interval of logsig is more comprehensive than $(-1, 1)$. tansig, controlling the active gap between $(-1, 1)$, ensures that the input range of each neural network layer is uniform. Such an activation function eliminates the need to discuss the input and hidden layers separately in subsequent work and also ensures the coherence of the BP neural network. This function helps to standardize the output range and effectively utilize the activation function's active interval. Tansig is suitable for evaluating the simulation data of the emergency braking function.

During the model training phase, the difference between the predicted values obtained via forward propagation and the true values needs to be measured using a loss function. The loss function calculates the discrepancy between the predicted and true values, allowing the derivation of loss values. In the backpropagation process, the model parameters are updated to minimize this error, allowing the predicted values to gradually converge to the true values and facilitate learning.

There are various forms of loss functions available, such as L1 parametric loss (L1 Loss), mean square error loss (MSE Loss), cross-entropy loss (CrossEntropy Loss), and KL divergence loss (KLDiv Loss), among others. These loss functions serve different purposes in model training objectives. MSE Loss is commonly used due to its smooth and continuous function curve. MSE also suits for using the gradient descent algorithm. Additionally, MSE Loss is convenient to compute gradients everywhere, making it faster to converge towards the minimum value even with a fixed learning rate. Therefore, in this paper, we select MSE as the loss function for the model. The formula is

$$MSE_i = \frac{1}{\sigma} \sum_{i=1}^{\sigma} (\hat{y}_i - y_i)^2 \tag{2}$$

$$\hat{y}_i = f_{act}(W_{ij} \times X_i + b_j) \tag{3}$$

In Equation (2) , $\hat{y}_i$ represents the predicted value, and y_i represents the true value. In Equation (3), f_{act} refers to the activation function.

Regarding vehicle braking scenarios, extensive research has been conducted by scholars both domestically and internationally. For instance, Kehtarnavaz et al. [15] introduced two parameters as inputs: the distance and angle between the main vehicle and the vehicle in front of it. They used the steering wheel angle and speed as outputs to construct a backpropagation (BP) neural network to accurately describe vehicle braking conditions and following distance.

Similarly, Wang et al. [16] utilized the expected acceleration and actual acceleration of the vehicle as inputs. The control acceleration of the vehicle dynamics model was used as the output to construct a three-layer BP neural network. This network was employed to develop an improved Proportional–Integral–Derivative (PID) controller for realizing the Autonomous Emergency Braking (AEB) function.

In this paper, after exporting the simulation parameters of the main vehicle, four parameters are selected as the input values of the neural network. These parameters

include the main vehicle's speed (v), the alarm time point (t_1), the control brake force at 40% time point (t_2), and the full braking time point (t_3). Consequently, the neural network has four nodes in the input layer. The output layer of the neural network consists of a single node. The output layer compares distance (d) [17] to the target car with the predicted value of the training output, when the main car completely brakes. Figure 4 provides a schematic representation of this setup.

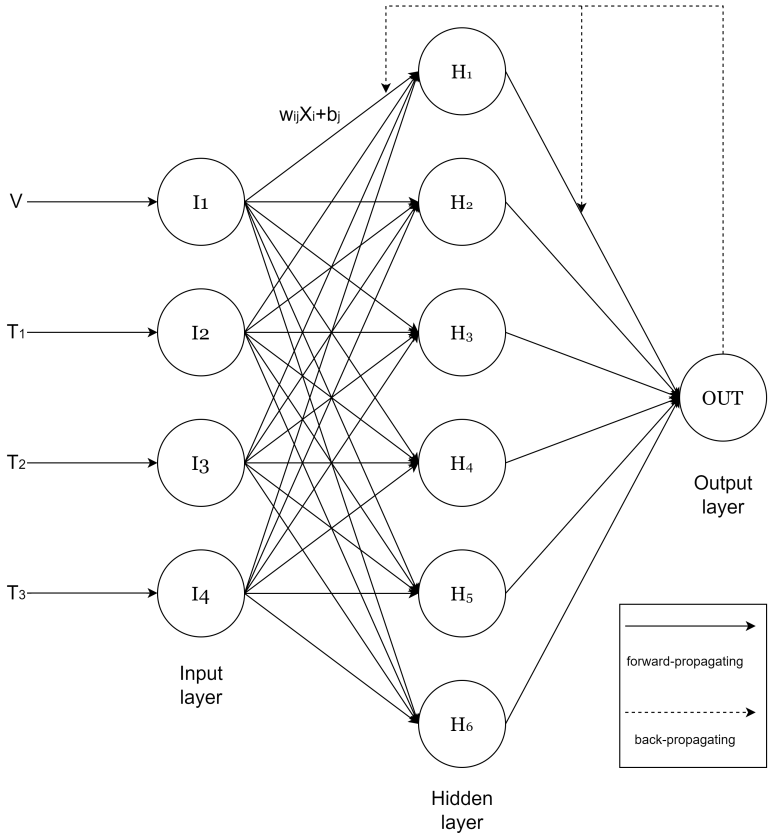

Figure 4. The structure diagram of BP neural network after determining the node.

After determining the number of nodes in the input layer and output layer, we must specify the number of nodes of the hidden layer neuron according to the number of input and output nodes. The number of hidden layer nodes is generally based on the trial method to obtain the selection range. Then, the best number is selected according to the influence of different neurons on the model during training. However, the trial method does not have stability in the actual process, and a more suitable approach is needed to determine the number of neurons; the formula is as follows. [18]

$$p < \sqrt{(m+n)} + d \tag{4}$$

The number of hidden layer nodes (p) is determined by the input (n) and output (m) nodes, incorporating a random constant (d) with a range of [0–10].

This specific example has four input parameters (n = 4) and a single output node (m = 1). Multiple training iterations have determined that the smallest training error is achieved when the number of hidden layer nodes is set to 6. As a result, this paper adopts an initial BP neural network model structure of 4-6-1. The neural network topology diagram (Figure 5) is depicted as follows:

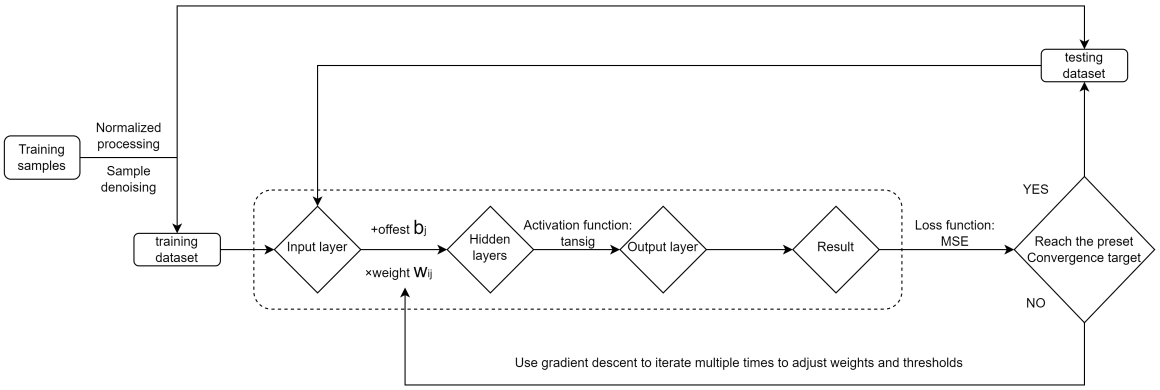

Figure 5. Neural network topology.

After determining the model of the BP neural network and the activation function, set the network parameters as follows: The number of training steps ≤1000, the learning rate is 0.1, and the convergence goal is 1×10^{-4} [19].

3.1.2. Improved BP Neural Networks

After the traditional BP neural network completes the forward propagation (input layer–hidden layer–output layer), it will perform backpropagation to optimize the network parameter configuration. Such a training process could also cause prediction errors of the test set to be minimized. The standard BP network uses the learning algorithm as the fastest descent method, also known as the gradient descent method. The use of the quickest descent method to adjust the weight of the network process is a process of reverse adjustment along the network, layer by layer, precisely to change the weights of the output layer and the hidden layer first. The weights between the hidden layer and the input layer are then adjusted. In the actual training process of using the fastest descent method, there will be a problem of slow convergence speed. In the optimization method of this problem, Ji Pengfei et al. [20] used the improved APSO adaptive particle swarm algorithm based on the particle swarm algorithm to optimize the neural network and adjust the global and local search and convergence capabilities by changing the particle inertia weight. Chen Junyi et al. [21] used the Dropout neural network model to obtain more accurate weighting results that met the criteria of expert evaluation. Huang et al. [22] used the improved SSA-BP neural network based on the sparrow algorithm to optimize the initial weights and thresholds of the BP neural network.

This paper chose to combine genetic algorithms to improve the initial weight parameters of neural networks.

3.2. GA-BP Neural Network

Genetic Algorithm Combined with the Roulette Selection Process

Genetic algorithms are computational models that emulate the biological evolutionary processes of natural selection and genetic mechanisms proposed by Darwin's theory of biological evolution. These algorithms simulate problem solving by incorporating concepts similar to the crossover and mutation of chromosome genes observed in biological evolution. Genetic algorithms have found extensive applications in various fields, such as combinatorial optimization, machine learning, signal processing, adaptive control, and artificial life.

The genetic algorithm has three stages: selection, crossover, and mutation. In the selection stage, this paper utilizes the roulette algorithm. In this method, the selection probability of each individual is determined based on its fitness value (fit). Individuals with higher fitness values are more likely to be selected for reproduction. However, in practice, the selection of individuals is often based on the "cumulative probability" rather than the individual's selection probability during roulette selection.

Firstly, the initial population is created by arranging the corresponding parameters of the initial neural network into a chromosome-like structure. Specifically, the weights corresponding to each group of samples are organized as the chromosomes of the population [23]. The initial population, made up of these chromosomes, is subject to genetic selection using the roulette algorithm.

After completing the initial population construction as shown in Figure 6, the parameters required for roulette are calculated using the following equation:

$$A_1 = f_{act}(W_1 \times X + b_1) \tag{5}$$

$$\hat{y} = w_2 \times A_1 + b_2 \tag{6}$$

$$fit_i = \frac{1}{MSE} = \frac{1}{\frac{1}{\sigma}\sum_{i=1}^{\sigma}(\hat{y}_i - y_i)^2} \tag{7}$$

In Equation (5), W_1 is the weight matrix formed after the full connection between the input layer and the hidden layer, There are a total of p*n weight parameters between the input layer and the hidden layer. X is the input data. f_{act} is the activation function of the neural network. In Equation (6), $\hat{y}$ is the prediction output data, b_1 is the weight bias after the input layer and the hidden layer are fully connected, b_2 is the weight deviation between the hidden layer and the output layer. In Equation (7), fit is a fitness function in the roulette selection method. It is worth noting that here, fit_i is the reciprocal of the mean squared error (MSE), and i is the number of current iterations, $i = 1, 2, \ldots \sigma$, σ is the population size.

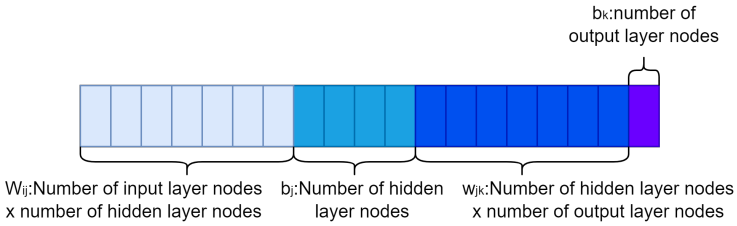

Figure 6. Schematic diagram of chromosomes in the genetic algorithm population.

There are many strategies to choose from in the traditional genetic algorithm selection phase. Traditional genetic algorithms use Monte Carlo selection, probabilistic selection, random traversal selection, and other methods.

Most researchers using genetic algorithms mostly use the the Monte Carlo selection process or linear sorting selection in the selection phase. Monte Carlo selection is the random selection of individuals in a given population. This selection method is arbitrary and non-generic.

When there is a significant difference in fitness among individuals, using the linear sorting selection method is more appropriate.

The individual fitness gap obtained after coding the ADAS simulation dataset is not large, and the linear sorting selection method is unsuitable. In order to improve the feedback accuracy and generalization of the model, the Monte Carlo selection method is also unsuitable for the optimized algorithm.

The authors considered the dataset elements and the kind of data that results from the ADAS function simulation. We incorporated the roulette-wheel algorithm into the genetic algorithm.

As shown in Figure 7, the probability of an individual being selected is related to the fitness value, and the higher the individual fitness value, the greater the probability of being selected. If the population number is M and the fitness of individual i is f_i, then the probability of individual i being selected via roulette selection is

$$P_i = \frac{f_i}{\sum_{M}^{k=1} f_k} \tag{8}$$

The function is continuously randomly selected to obtain a new batch of optimized populations.

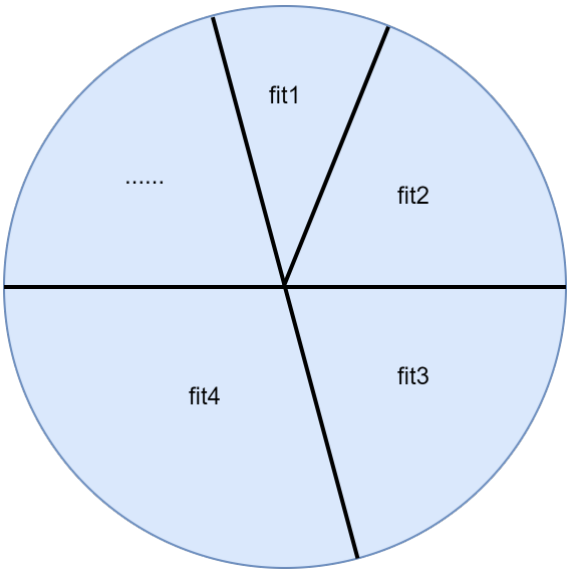

Figure 7. Schematic diagram of the roulette selection process.

This is followed by the crossover and mutation phases, which, in genetic algorithms, essentially increase the diversity of the population, and then the optimal adaptive population is selected in the iterative process.

Cross-manipulation: In the genetic algorithm, selected individuals undergo a genetic exchange to produce the next generation. The probability of crossover, known as the crossover probability, determines the likelihood of individuals experiencing crossover. The general range for the crossover probability is 0.4 to 0.99, while a value of 0.4 is used in this article. Various crossover methods are available, including single-point/multi-point crossover, uniform crossover, and arithmetic crossover. This paper employs the two-point crossover method, where segments between randomly selected intersection points of paired chromosomes are swapped. Figure 8 illustrates this process, where specific segments within the circular chromosome are exchanged following the placement of intersection points.

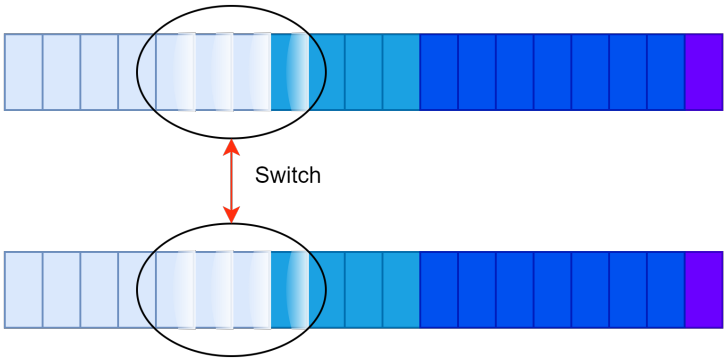

Figure 8. Cross-operation process.

Mutation operation: After the crossover process, individuals are subjected to mutation with a certain probability, creating new mutated individuals. In practical applications, a commonly used method is single-point variation, also known as bit variation. Only one bit in the gene sequence is mutated in this method, as shown in Figure 9. The mutation probability determines the likelihood of mutation occurring. Typically, the mutation

probability ranges from 0.001 to 0.1 [24]. In this paper, we utilize a mutation probability of 0.05.

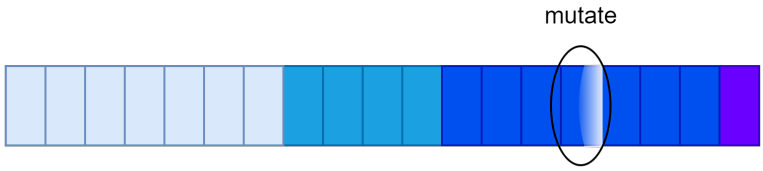

Figure 9. Mutation operation process.

As shown in Figure 10, the modified GA-BP neural network is divided into a training part and a genetic optimization part [25].

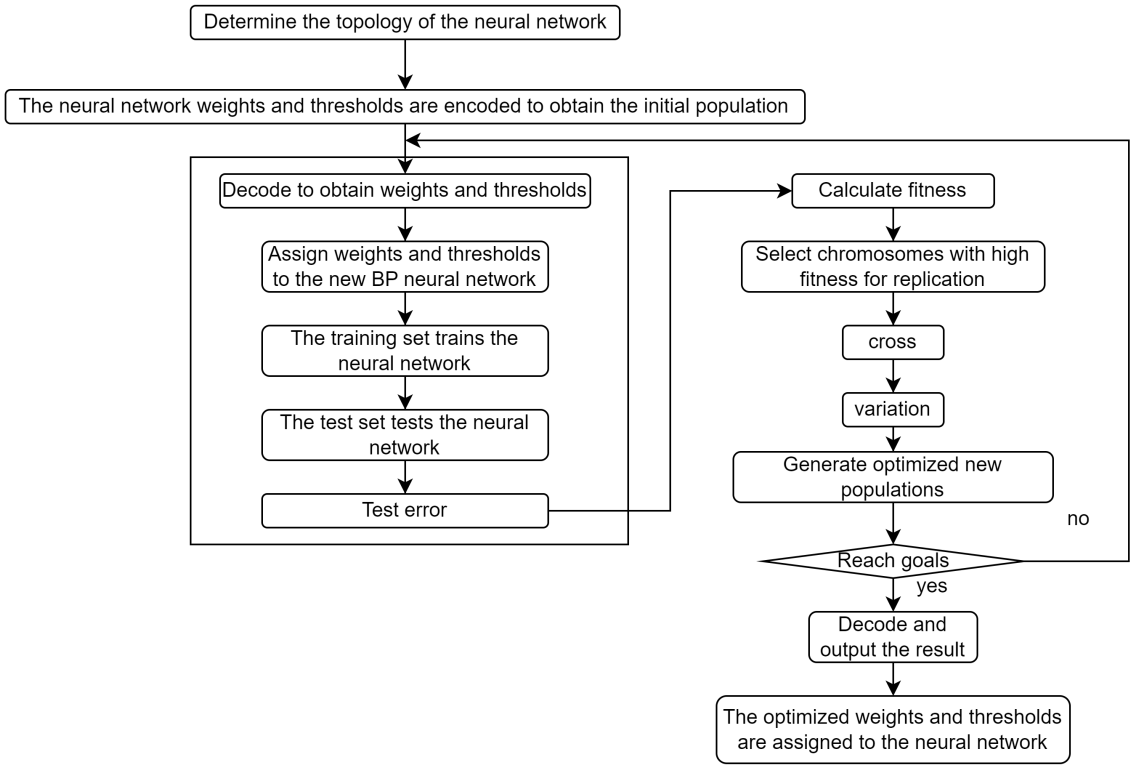

Figure 10. Flow chart of improved GA-BP neural network.

3.3. Sample Data Preprocessing

Before the simulation result data is entered into the neural network for training, the sample data needs to be preprocessed to remove some outliers, and the outliers are judged by this equation:

$$MAD = median(|X_i - median(X)|) \tag{9}$$

In Equation (9), MAD represents the absolute $median$, and $median(X)$ represents the sample median. If we assume that the sample follows a normal distribution, the range of outliers is set to 50% of the area on both sides of the distribution. We can determine the range of outliers according to the following equation:

$$\begin{gathered} P(X - \mu) \leq MAD \\ P(\frac{X - \mu}{\sigma} \leq \frac{MAD}{\sigma}) = P(Z \leq \frac{MAD}{\sigma}) = \frac{1}{2} \end{gathered} \tag{10}$$

Because

$$\Phi(-\alpha) = 1 - \Phi(\alpha) \tag{11}$$

So

$$\Phi(-\frac{MAD}{\sigma}) = \frac{3}{4} \tag{12}$$

The researcher check the table and knew $MAD/\sigma = 0.675$. Then, it can be derived that when

$$\begin{cases} X \geq 1.4826MAD \\ X \leq -1.4826MAD \end{cases} \tag{13}$$

X is an outlier when it is in the 50% region on both sides of the distribution.

After the above preprocessing, 28 sets of outliers were excluded, where 140 sets of the remaining 172 sets of data were taken as the training set, and the rest were the test set.

4. Results

Comparison of Training Results

Figure 11 represents the fitness change curve obtained by applying the genetic algorithm to optimize the BP neural network. From this figure, Figure 11, it can be observed that both the optimal population and the average fitness level tend to stabilize after the 45th generation. Therefore, we can conclude that the genetic algorithm reaches its optimal state by the 45th generation. The best chromosome obtained after the 45th generation is decoded and assigned as the initial set of parameters for the improved BP neural network. Subsequently, the gradient descent method is employed to search for the minimum value locally, enabling the training of the network to achieve optimization.

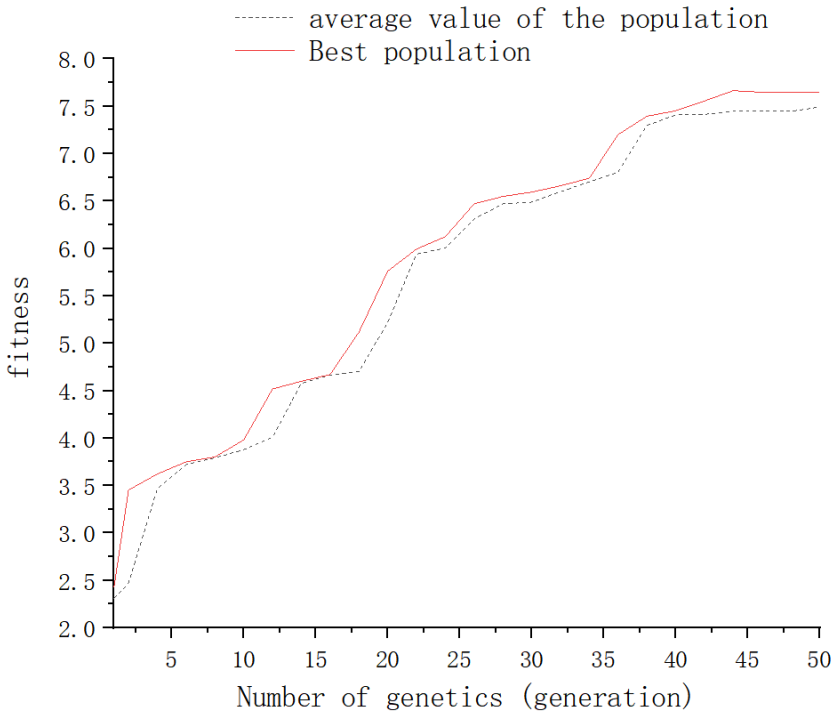

Figure 11. The result of changes in fitness with genetic algebra.

Based on the analysis of Figure 12, a conclusion can be drawn that the convergence speed of BP neural networks optimized via genetic algorithms is considerably faster compared to regular BP neural networks [26]. The optimized BP neural networks achieve convergence within 250–300 iterations, while traditional BP neural networks typically require 300–400 iterations to converge gradually. Moreover, the optimized BP neural net-

works based on genetic algorithms achieve better minimum values for the mean square error (MSE) compared to regular BP neural networks [27,28].

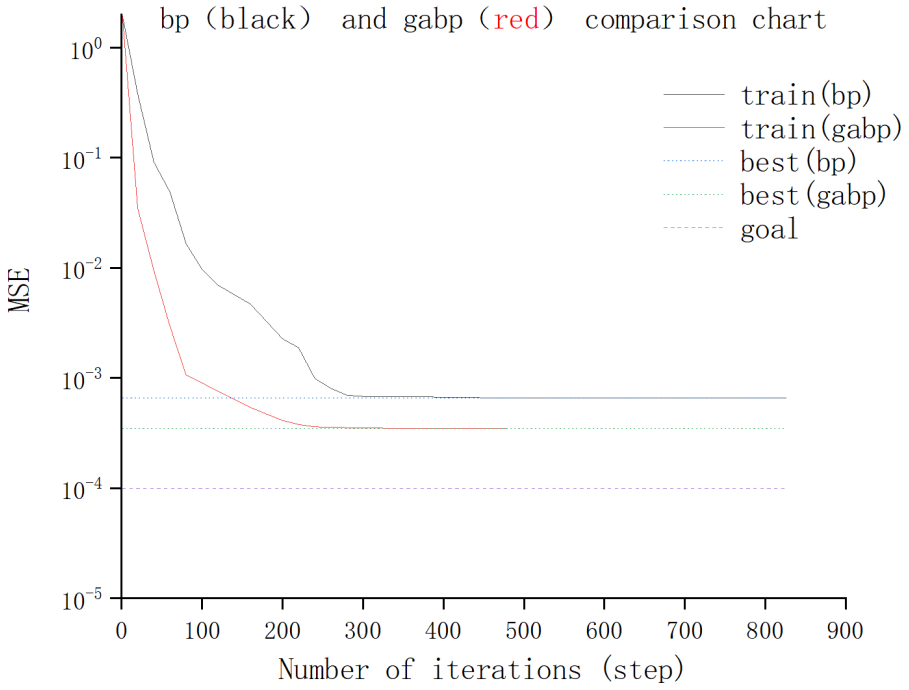

Figure 12. BP and GA-BP neural networks' training results comparison chart.

The test dataset is fed into both the regular BP neural network and the optimized BP neural network. The predicted values of the two different neural networks are then compared with the actual values from the test dataset. This comparison allows for evaluating the performance and accuracy of the regular BP neural network and the optimized BP neural network in predicting the target values.

From Figure 13, it is evident that both the regular BP neural network and the improved GA-BP neural network can fit the real values reasonably well. However, upon closer examination, it becomes clear that the GA-BP neural network exhibits superior tracking of the actual values compared to the regular BP neural network. Additionally, the average error between the predicted values and the actual values of the GA-BP neural network is smaller than that of the regular BP neural network.

To further analyze the model's prediction bias, the performance of the models is evaluated using regression evaluation metrics. Table 2 presents the values of 32 test sets alongside the corresponding prediction results. Table 3, on the other hand, compares the regression evaluation metrics for the errors of the two neural networks.

It can be seen from Table 3 that the prediction error of the GA-BP neural network is better than that of the ordinary BP neural network, which is reduced by 25.83%, 13.89%, and 17.13% by MSE, RMSE, and MAE, respectively. The coefficient of determination R_2 (R-Square) is a regression evaluation index that judges the prediction results of neural network models [29], and its formula is

$$R_2 = 1 - \frac{\sum_{i=1}^{m}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{m}\bar{y}_i - y_i)^2} \tag{14}$$

In Equation (14), i is the test data serial number, and $i = 1, 2, 3 \ldots, m$, m is the test set sample size. As shown in Equation (14), the value range of R_2 is [0, 1], and in general, the larger R_2, the better the model fit. Moreover, the size of R_2 is related to the sample size, which can roughly quantitatively evaluate the accuracy of the model and also explain the generalization ability of the model prediction. After the genetic algorithm optimized the BP neural network, R_2 was improved by 0.4% compared with the original neural network.

According to the results of the above regression evaluation indicators, we can find that after optimizing the original BP neural network using the genetic algorithm, this paper has better generalization ability and prediction ability and can better predict the distance from the target vehicle after the AEB function is triggered and completely stopped.

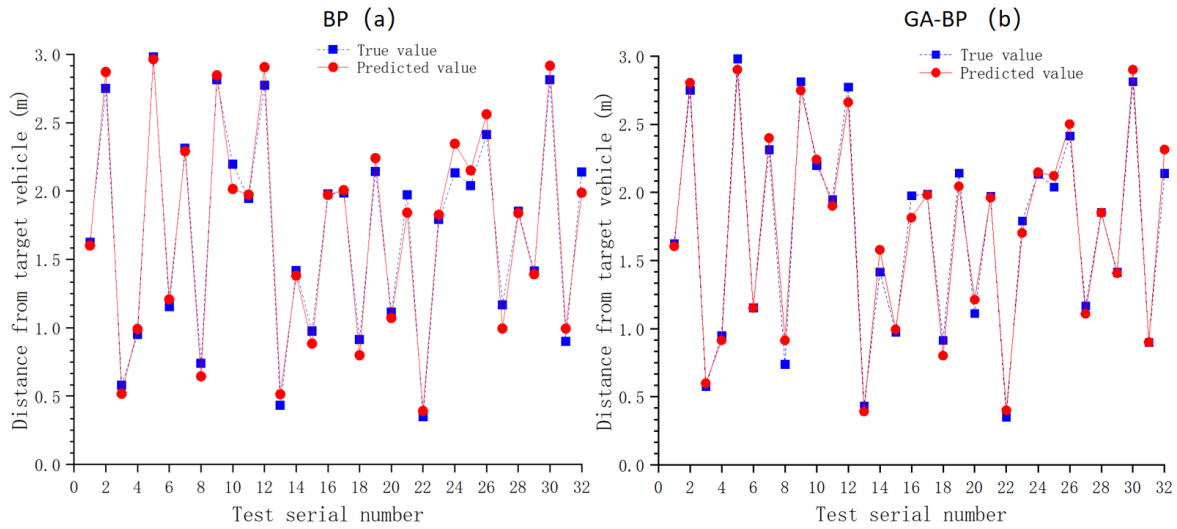

Figure 13. BP neural network prediction results (**a**); modified GA-BP neural network prediction results (**b**).

Table 2. Some test set parameters and prediction results.

Serial Number	Index	Value	Symbol	Dimension
1	Minimum self-driving speed	46.24	V_{min}	km/h
2	Maximum self-driving speed	97.51	V_{max}	km/h
3	The earliest time point of alarm	4.22	T_{1ear}	s
4	The latest time point of alarm	6.91	T_{1lat}	s
5	The earliest time point of 40% braking force	4.97	T_{2ear}	s
6	The latest time point of 40% braking force	7.84	T_{2lat}	s
7	The earliest time point of full-force braking	5.03	T_{3ear}	s
8	The latest time point of full-force braking	8.37	T_{3lat}	s
9	Maximum distance after actual braking	2.981	d_{max}	m
10	Minimum distance after actual braking	0.351	d_{min}	m
11	Average vehicle distance of vehicle after actual braking	1.712	$\bar{d}$	m
12	Maximum distance after predicted braking	2.901	$\hat{d}_{max}$	m
13	Minimum distance after predicted braking	0.391	$\hat{d}_{min}$	m
14	Average vehicle distance of vehicle after predicted braking	1.718	$\hat{\bar{d}}$	m

Table 3. Comparison of regression evaluation indicators of two neural network errors.

	Mean Squared Error	Root Mean Square Error	Mean Absolute Error	Coefficients of Determination (R2)
BP	9.395×10^{-2}	9.693×10^{-2}	7.934×10^{-2}	9.829×10^{-1}
GA-BP	6.968×10^{-2}	8.347×10^{-2}	6.575×10^{-2}	9.873×10^{-1}
Optimization volume	25.83%	13.89%	17.13%	0.4%

The author implemented the aforementioned enhanced algorithms (APSO-BP [30], PSO-BP [31,32], and SSA-BP [33,34]) and applied the same regression evaluation metrics to compare them with the GA-BP neural network. The results of this comparison are as follows (Table 4):

Table 4. Comparison of the results of the modified GA-BP algorithm with other algorithms.

	Mean Squared Error	Root Mean Square Error	Mean Absolute Error	Coefficients of Determination (R2)
APSO-BP	8.423×10^{-2}	9.041×10^{-2}	7.514×10^{-2}	9.846×10^{-1}
PSO-BP	8.672×10^{-2}	8.945×10^{-2}	7.044×10^{-2}	9.837×10^{-1}
SSA-BP	9.071×10^{-2}	8.794×10^{-2}	6.949×10^{-2}	9.840×10^{-1}

The regression evaluation indicators demonstrate that all three improved BP neural networks have improved in accuracy compared to the original neural network. However, the improvement is inferior to the GA-BP neural network proposed in this paper.

5. Conclusions

Based on the functional simulation test data of AEB, this paper initially builds an AEB evaluation model using a regular BP neural network. Subsequently, the standard BP neural network is optimized using a genetic algorithm based on the roulette selection process. The experimental results demonstrate that the improved GA-BP neural network outperforms the original BP neural network regarding prediction error and regression evaluation metrics. This enhancement effectively enhances the prediction accuracy and robustness of the neural network. Consequently, the GA-BP neural network better assesses the AEB algorithm's performance in the simulation software.

This study evaluates and predicts AEB function simulation data. AEB belongs to the basic of ADAS functions. There are many other complex ADAS functions. In subsequent studies, we will conduct similar evaluation and prediction of other ADAS functions.

In addition, research on ADAS simulation scenarios is yet to be conducted. Since simulation scenarios directly affect the final simulation data, we should develop specific scenarios based on the actual situation and the function under test.

Some more cutting-edge concepts, such as SOTIF (Safety of the Intended Functionality), should be added to the scenario development process [35]. In order to avoid potential safety hazards, scenario awareness is required so that the automated system can make decisions based on the environment. We will continue to explore this area.

Author Contributions: Conceptualization, S.Z.; methodology, S.Z.; software, S.Z.; validation, L.C.; formal analysis, S.Z.; investigation, S.Z.; resources, Y.H.; data curation, S.Z.; writing—original draft preparation, S.Z.; writing—review and editing, S.Z.; visualization, S.Z.; supervision, L.C.; project administration, Y.H.; funding acquisition, Y.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Shanghai Municipal Commission of Science and Technology, and the project name "Scenario Factory, a working platform for the generation and processing of autonomous driving simulation scenarios". The grant number 220H1053500.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

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

References

1. Yu, W.-Z.; Su, Y.-M.; Wang, L. A Review of Research on Evaluation of Automated Driving Tests. *Syst. Sci. Math.* **2022**, *42*, 495–508. (In Chinese)
2. Shaout, A.; Colella, D.; Awad, S. Advanced driver assistance systems-past, present and future. In Proceedings of the 2011 Seventh International Computer Engineering Conference (ICENCO'2011), Cairo, Egypt, 27–28 December 2011; pp. 72–82.
3. Ming, C.-L.; Rong, W.-S. Overview of Vehicle AEB. *Mod. Electron. Technol.* **2021**, *5*, 14–16.
4. El Mostadi, M.; Waeselynck, H.; Gabriel, J.M. Virtual Test Scenarios for ADAS: Distance to Real Scenarios Matters! In Proceedings of the 2022 IEEE Intelligent Vehicles Symposium (IV), Aachen, Germany, 4–9 June 2022; pp. 836–841.

5. Wu, Z.; Yang, J.; Huang, L. Study on the collision avoidance strategy at unsignalized intersection based on PreScan simulation. *Procedia-Soc. Behav. Sci.* **2013**, *96*, 1315–1321. [CrossRef]
6. Guo, L.-B.; Mei, Z.-Q.; He, Y. Simulation of Intersection Collision Avoidance Control System Based on PreScan Software. *Mech. Electron.* **2014**, *2*, 22–26. (In Chinese)
7. Zhang, Q.; Chen, D.; Li, Y.; Li, K. Research on performance test method of lane departure warning system with PreScan. In *Proceedings of SAE-China Congress*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 445–453.
8. Yang, W.; Liu, J.; Zhou, K.; Zhang, Z.; Qu, X. An automatic emergency braking model considering driver's intention recognition of the front vehicle. *J. Adv. Transp.* **2020**, *2020*, 5172305. [CrossRef]
9. Kim, B.; Kwon, B.; Lee, S. A study on evaluation method of AEB pedestrian test. *J. -Auto-Veh. Saf. Assoc.* **2018**, *10*, 25–32.
10. Tideman, M.; Van Noort, M. A simulation tool suite for developing connected vehicle systems. In Proceedings of the 2013 IEEE Intelligent Vehicles Symposium (IV), Sofitel Broadbeach, Gold Coast, Australia, 23–26 June 2013; pp. 713–718.
11. Gao, Y.; Xu, Z.; Zhao, X.; Wang, G.; Yuan, Q. Hardware-in-the-loop simulation platform for autonomous vehicle AEB prototyping and validation. In Proceedings of the 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), Rhodes, Greece, 20 September 2020; pp. 1–6.
12. Kim, J.; Han, H.; Kim, S.; Choi, J.; Park, J.; Park, H. A study on the AEB operation simulation using prescan based on the vehicle test. *Trans. Korean Soc. Automot. Eng.* **2020**, *30*, 249–257. [CrossRef]
13. Poggio, T.; Girosi, F. Regularization algorithms for learning that are equivalent to multilayer networks. *Science* **1990**, *247*, 978–982. [CrossRef]
14. Ding, S.; Su, C.; Yu, J. An optimizing BP neural network algorithm based on genetic algorithm. *Artif. Intell. Rev.* **2011**, *36*, 153–162. [CrossRef]
15. Kehtarnavaz, N.; Groswold, N.; Miller, K.; Lascoe, P. A transportable neural-network approach to autonomous vehicle following. *IEEE Trans. Veh. Technol.* **1998**, *47*, 694–702. [CrossRef]
16. Wang, L.; Zhan, Z.; Yang, X.; Wang, Q.; Zhang, Y.; Zheng, L.; Guo, G. Development of BP neural network PID controller and its application on autonomous emergency braking system. In Proceedings of the 2018 IEEE Intelligent Vehicles Symposium (IV), Changshu, China, 26–30 June 2018; pp. 1711–1716.
17. Han, J.; Heo, O.; Park, M.; Kee, S.; Sunwoo, M. Vehicle distance estimation using a mono-camera for FCW/AEB systems. *Int. J. Automot. Technol.* **2016**, *17*, 483–491. [CrossRef]
18. Suliman, A.; Zhang, Y. A review on back-propagation neural networks in the application of remote sensing image classification. *J. Earth Sci. Eng.* **2015**, *5*, 52–65.
19. Zhao, Z.; Xin, H.; Ren, Y.; Guo, X. Application and comparison of BP neural network algorithm in MATLAB. In Proceedings of the 2010 International Conference on Measuring Technology and Mechatronics Automation (Vol.1), Changsha, China, 13–14 March 2010; pp. 590–593.
20. Ji, P.-F.; Meng, W.-N.; Yang, B.-F.; Wang, D.-D. Research on Agricultural Machinery Data Prediction Based on Improved BP Neural Networks. *China Agrochem. J.* **2020**, *41*, 200–205. (In Chinese)
21. Chen, J.-Y.; Chen, L.; Meng, H.-L.; Xiong, L. A neural network-based evaluation model for vehicle traffic coordination. *J. Tongji Univ. (Nat. Sci. Ed.)* **2021**, *49*, 135–141. (In Chinese)
22. Huang, Y.; Luo, W.; Lan, H. Adaptive pre-aim control of driverless vehicle path tracking based on a SSA-BP neural network. *World Electr. Veh. J.* **2022**, *13*, 55. [CrossRef]
23. Guo, Y.-J.; Lu, Y.-P.; Zhang, Y.-H.; Wang, C. Optimization of BP neural network car-following model. *Chin. J. Saf. Sci.* **2016**, *26*, 103–108. (In Chinese)
24. Wang, T.-F.; Liu, X.-Y. Research on Vehicle Following Model Based on BP Neural Networks. *South. Agric. Mach.* **2023**, *54*, 20–23. (In Chinese)
25. Zhu, L.; Zhang, S.; Xu, S.; Zhao, H.; Chen, S.; Wei, D.; Liu, J. Classification of UAV-to-ground targets based on micro-Doppler fractal features using IEEMD and GA-BP neural network. *IEEE Sens. J.* **2019**, *20*, 348–358. [CrossRef]
26. Qian, K.; Hou, Z.; Sun, D. Sound quality estimation of electric vehicles based on GA-BP artificial neural networks. *Appl. Sci.* **2019**, *10*, 5567. [CrossRef]
27. Liao, M.; Liang, S.; Luo, R.; Chen, Y. The moving load identification method on asphalt roads based on the BP neural network and FBG sensor monitoring. *Constr. Build. Mater.* **2023**, *378*, 131216. [CrossRef]
28. Tang, D.; Zhang, Z.; Hua, L.; Pan, J.; Xiao, Y. Prediction of cold start emissions for hybrid electric vehicles based on genetic algorithms and neural networks. *J. Clean. Prod.* **2023**, *420*, 131216. [CrossRef]
29. Akossou, A.Y.J.; Palm, R. Impact of data structure on the estimators R-square and adjusted R-square in linear regression. *Int. J. Math. Comput.* **2013**, *20*, 84–93.
30. Sun, Y.; He, J.-B.; Gao, Y.-L. Application of APSO-BP Neural Network Algorithm in Stock Price Prediction. In Proceedings of the 14th International Conference, ICSI 2023, Proceedings, Part II, Shenzhen, China, 14–18 July 2023; pp. 478–489.
31. Tang, X.; Shi, L.; Wang, B.; Cheng, A. Weight adaptive path tracking control for autonomous vehicles based on PSO-BP neural network. *Sensors* **2022**, *23*, 412. [CrossRef] [PubMed]
32. Wu, J.; Cui, J.; Shu, Y.; Wang, Y.; Chen, R.; Wang, L. Multi-level health degree analysis of vehicle transmission system based on PSO-BP neural network data fusion. *Eksploat. Niezawodn.* **2023**, *25*, 4. [CrossRef]

33. Zheng, Y.; Li, L.; Qian, L.; Cheng, B.; Hou, W.; Zhuang, Y. Adaptive Leakage Protection for Low-Voltage Distribution Systems Based on SSA-BP Neural Network. *Appl. Sci.* **2023**, *13*, 9273.
34. Liu, Z.; Yu, H.; Jin, W. Sine-SSA-BP Ship Trajectory Prediction Based on Chaotic Mapping Improved Sparrow Search Algorithm. *Sensors* **2023**, *23*, 704.
35. Khatun, M.; Glaß, M.; Jung, R. Scenario-based extended HARA incorporating functional safety & SOTIF for autonomous driving. In Proceedings of the ESREL-30th European Safety and Reliability Conference, Venice, Italy, 1–5 November 2020.