Vehicle Feature Recognition via A Convolutional Neural Network with An Improved Bird Swarm Algorithm

Xuan Chen[*]

Design and Art Branch, Zhejiang Industry Polytechnic College, China
chenxuan1979@sina.com

Abstract

Accurate vehicle feature recognition is an important element in traffic intelligence systems. To address the problems of slow convergence and weak generalization ability in using convolutional neural networks to improve vehicle feature recognition, we propose an improved bird swarm algorithm to optimize convolutional neural networks (IBSA-CNNs) for vehicle recognition strategies. First, we use the center of gravity backward learning strategy and similarity- and aggregation-based optimization strategy in population initialization and foraging behavior, respectively, to improve the algorithm performance and avoid falling into a local optimum. Second, the improved bird swarm algorithm is used to optimize the weights of the convolutional and pooling layers of the convolutional neural network to improve the neural network performance. Finally, we tested the performance of the improved bird swarm algorithm in simulation experiments using benchmark functions. The recognition performance of IBSA-CNN was tested by the UCI dataset, and in the traffic vehicle dataset BIT-Vehicle, it improved 4.9% and 6.8% compared with R-CNN and CNN, respectively, indicating that IBSA-CNN has better vehicle feature recognition.

Keywords: Bird swarm algorithm, Convolutional neural network, Feature recognition

1 Introduction

Intelligent transportation systems [1] (ITSs) have become an important part of traffic information management in various countries, and accurate vehicle identification can help the traffic portion of vehicle management. Vehicle recognition refers to vehicle image information that contains the relevant characteristic information of the vehicle by means of efficient acquisition equipment. The vehicle image processing technology locates and extracts the characteristics of the image vehicle target. Traditional image feature recognition techniques such as laser detection, infrared detection, radar detection, etc., have been unable to adapt to the current vehicle detection requirements [2], and neural network models with low cost and obvious performance advantages have been widely used in vehicle recognition. Convolutional neural network (CNN) [3] is an effective recognition model, which has the advantages of simple structure and superior performance. However, the setting of the weight values of the convolutional and pooling layers composing this network leads to problems leading to local optimum, slow convergence and weak generalization ability of this model, so the reasonable optimization of the weights of the convolutional neural network becomes a direction for solving the convolutional neural network. Because the metaheuristic optimization algorithm does not require gradient information, the number of parameters to be adjusted is small, and its own convergence speed is fast in low-dimensional conditions, we propose to optimize the convolutional neural network strategy by the metaheuristic optimization algorithm. The bird swarm algorithm [4] (BSA) is a swarm intelligence algorithm based on bird swarm behavior proposed by Xian-Bing Meng in 2015. The algorithm was inspired by the study of bird swarm behavior, and it was constructed by imitating the foraging, alerting and flying behaviors that occur in bird swarm predation. It obtains the optimal solution based on the information sharing mechanism and search strategy. To address the drawbacks of slow convergence and the tendency to fall into a local optimum when solving high-dimensional problems, we optimize the bird swarm algorithm in the following two aspects: in the initialization of the population, we use the center of gravity reverse learning strategy to improve the diversity of individuals in the population and enrich the number of understandings; and in the foraging behavior, we use the optimization strategy based on similarity and aggregation to improve the search ability of the swarm individuals and avoid the algorithm falling into a local optimum. We use the improved bird swarm algorithm for weight optimization of CNN. The simulation experimental process tests the performance of IBSA algorithm and the recognition performance of IBSA-CNN in UCI dataset. Finally, we compare the recognition effect of this model with RNN and regional convolutional neural network (R-CNN) in the vehicle feature database BIT-Vehicle dataset, and the results show that this model has good recognition effect.

To further elaborate the implementation strategy, we structure the paper as follows: Section 2 describes the current research on vehicle recognition, Section 3 illustrates the principles of the bird swarm algorithm and convolutional neural network, Section 4 illustrates the convolutional neural network model based on the improved bird swarm algorithm, Section 5 illustrates the simulation experimental process to verify the effectiveness of the proposed model, and Section 6 presents to conclusions of the paper.

*Corresponding Author: Xuan Chen; E-mail: chenxuan1979@sina.com

DOI: 10.53106/160792642023032402020

2 Related Knowledge

Vehicle feature recognition has received major attention from scholars' in various countries, mainly including feature recognition based on a priori knowledge recognition, feature training recognition based on features and feature recognition based on artificial intelligence techniques.

In prior knowledge-based recognition, the candidate regions of the target are generated based on the prior knowledge of the target to be recognized, such as symmetric features [5], grayscale features [6] and other information inherent to the vehicle's own feature attributes. Otsu [7] uses the threshold value automatically calculated by the maximum interclass variance method to filter out the possible location regions of vehicle taillights and then uses the HSV space to filter the nontaillight regions to complete the vehicle target recognition. Tan et al. [8] decides the pose and category of the target vehicle in the form of one-dimension with voting for the image gradient information in the traffic scene, which improves the effectiveness and robustness of recognition. Fung et al. [9] proposed estimating the 3D data of the target using monocular image sequences and the morphology of the vehicle using a four-stage state approximation method. The results illustrate that the vehicle area obtained by this method is almost the same as the actual size.

In feature-based training recognition methods, machine learning strategies are used because they can extract the implicit features and patterns in the input data, such as histograms of oriented gradients [10], Gabor features [11], support vector machines [12], the Waldboost method [13] and the AdaBoost method [14]. Leyrit et al. [15] proposed using target contour and texture instead of class Haar features to obtain deep-level feature information of samples by simple information description. Khammari et al. [16] proposed a gradient-based method and AdaBoost classification for vehicle detection and experimentally verified that it has better results when tested under different traffic scenarios. Broggi et al. [17] proposed an AdaBoost method with Haar feature decision tree stump composition of AdaBoost classifier in vehicle detection, and experiments illustrate that the strategy has better results in vehicle recognition on urban roads and high-speed roads. Sun et al. [18] uses a set of Gabor filters optimized specifically for vehicle detection tasks to improve the performance of road vehicle detection. Their experiments illustrate that the method has better results.

In artificial intelligence techniques, Collins et al. [19] used neural networks in the field of vehicle recognition. Dewangan et al. [20] used vehicle images to train unsupervised neural networks to achieve better results in classifying vehicles. Maity et al. [21] proposed a better guide to use the R-CNN model on vehicle recognition. Charouh et al. [22] proposed a framework to reduce the complexity of CNN-based vehicle recognition methods by optimizing the number of convolutional operations of CNN through Background Subtraction, and simulation experiments illustrated that it can effectively reduce the computation of CNN models in scale vehicle dataset testing. Luo et al. [23] proposed a faster R-CNN based model, which combines NAS optimization and feature enrichment to achieve effective detection of multiscale vehicle targets in traffic scenes. Hu et al. [24] proposed a cascaded vehicle detection method with a multifeature fusion convolutional neural network, which extracts the features of local binary pattern of front vehicle, Haarlike and directional gradient histogram, and then performs principal component analysis downscaling and serial fusion on the input image, and simulation experiments illustrate the good robustness of the method. Ghosh et al. [25] used faster R-CNN with multiple region proposal networks (RPN) to detect and track road vehicles under different weather conditions. The ROI in this method is generated using several RPNs of different sizes, so it is able to detect vehicles of different sizes, and simulation experiments illustrate that this algorithm has advantages. Karungaru et al. [26] combines CNN with SVM and normalizes the features in SVM to improve the generalization ability of the model, and achieves better results in vehicle detection and classification in real road environments. Anandhalli et al. [27] proposes a method using TensorFlow fusion for vehicle detection and tracking. Their simulation experiments illustrate that the proposed algorithm has results of 90.88% detection accuracy. Soon et al. [28] proposed a strategy to optimize CNN parameters using particle swarm algorithm to minimize the variability in vehicle recognition training. Simulation experiments illustrated that the method has a good effect and the recognition rate reached 99.1%. Fu et al. [29] proposed a multiscale integrated feature fusion convolutional neural network (MCFF-CNN) based on residual learning for vehicle feature recognition. Their simulation experiments illustrate the good effect of the MCFF-CNN for vehicle color recognition in real traffic scenarios. Hsu et al. [30] proposes a simplified fast region-based convolutional neural network for vehicle detection; the neural network demonstrates good recognition effect in VTTI database. Kukreja et al. [31] proposes the use of a generative adversarial network algorithm to correct the problem of low accuracy of CNN and RNN in recognizing license plate numbers under noise interference. The simulation experiments illustrate that the algorithm achieves 99.39% accuracy in the recognition of license plates. Tourani et al. [32] proposes a method for vehicle detection based on CNN for video frames. The experimental results show that it takes only 74 ms on average to detect vehicles in real data, which has a good detection effect in terms of accuracy and execution time. Huang et al. [33] proposes a method for the recognition of vehicle manufacturer criteria using convolutional neural networks. Their simulation experiments illustrate that the method obtains a relatively high recognition rate. Kaur et al. [34] proposes an efficient system for license plate character recognition using CNN, and experiments illustrate that this proposed technique achieves an overall accuracy of 98.13%. Zhang et al. [35] proposes a new multiview convolutional neural network model (MV-CNN) method for vehicle driving behavior recognition, which is used in the behavior recognition of vehicle driving with six-axis motion sensors. Their experiments show that the method has a high recognition rate. There also exist some works about the neural architecture research in order to figure out the neural networks with the best performance [36-38]

Reviewing these research results, we found that the use of various machine learning neural network algorithms has become the current research direction of scholars, so maximizing the function of neural networks for vehicle feature recognition has become an important direction of current research in the field of traffic intelligence. We start our research based on this direction.

3 Basic Algorithm

3.1 Bird Swarm Algorithm

Xian-Bing Meng's algorithm is divided into the three components of foraging behavior, vigilance behavior and flight behavior according to the behavioral characteristics of the swarm. Therefore, we set the population size of the swarm algorithm in this paper as N and the search space as the D dimension, so the position of the i th bird ($i \in [1,2,...,N]$) individual in the E dimension space is denoted as $X_i = (X_i^1, X_i^2, ..., X_i^D)$.

(1) Foraging behavior

Each bird foraged with its own experience and that of the whole population, and individual positions were updated as follows.

$$x_{i,j}^{t+1} = x_{i,j}^t + (p_{i,j} - x_{i,j}^t) \times C \times rand + (g_j - x_{i,j}^t) \times S \times rand. \quad (1)$$

where $x_{i,j}^{t+1}$ and $x_{i,j}^t$ denote the position of the individual in the $t+1$ and t iterations of the i th bird in the j th dimension, respectively $rand$ denotes a random number between (0,1), C and S are the perceptual and social driving coefficients, respectively $p_{i,j}$ denotes the best position of the i th bird in the j th dimension, and g_j denotes the best position of the whole population in the j th dimension.

(2) Vigilance behavior

In the process of flight, each individual bird is trying to move to the center of the whole population. In the process of moving, there will be inevitable competition and hindrance between individuals, so we set this behavior as alert behavior, and the individual position update formula under this behavior is as follows:

$$\begin{aligned} x_{i,j}^{t+1} = x_{i,j}^t &+ A_1 \times (mean_j - x_{i,j}^t) \times rand \\ &+ A_2 \times (p_{k,j} - x_{i,j}^t) \times rand, \end{aligned} \quad (2)$$

$$A_1 = a_1 \times \exp(-\frac{pFit_i}{sumFit + \varepsilon} \times N), \quad (3)$$

$$A_2 = a_2 \times \exp((\frac{pFit_i - pFit_k}{| pFit_k - pFit_i | + \varepsilon}) \frac{N \times pFit_k}{sumFit + \varepsilon}), \quad (4)$$

where $mean_j$ denotes the mean position in dimension j, $p_{k,j}$ denotes the position of the k th bird in dimension j, and k is a random positive integer between $[1, N]$ and $k \neq i$. a_1 and a_2 are constants between [0,2], $pFit_i$ and $pFit_k$ denote the best fit values of the i th and k th birds, respectively $sumFit$ denotes the sum of the best fit values of the whole population, and ε is a small constant with the denominator avoiding zero. A_1 denotes the indirect effect caused by the surrounding environment when the bird moves toward the population center, and A_2 denotes the direct effect caused by the specific disturbance when the bird moves toward the population center. When the fitness of individual k is better than the fitness of individual i, it means that individual i suffers from greater disturbance than individual k, and therefore individual k may also move toward the population center.

(3) Flight behavior

During the flight, the swarm will fly to other places due to predation or other disturbances from outside, and when looking for food again, some individual birds will play the role of producers to find food, while others may play the role of beggars to follow the producers. Therefore, producers and beggars are defined as follows:

$$x_{i,j}^{t+1} = x_{i,j}^t + randn(0,1) \times x_{i,j}^t, \quad (5)$$

$$x_{i,j}^{t+1} = x_{i,j}^t + (x_{k,j}^t - x_{i,j}^t) \times FL \times rand, \quad (6)$$

where $randn(0,1)$ is a Gaussian distribution with mean 0 and standard deviation 1. $k \in [1, N]$, $k \neq i$, and $FL \in (0,2]$ are used to indicate that the beggar follows the producer in need of food, and the frequency of birds flying to other places is set to FL and is a positive integer.

In Equation (5), we use the Gaussian distribution to express the characteristics of the flight behavior of the swarm; that is, $randn(0,1)$ denotes the Gaussian distribution with mean 0 and standard deviation 1, $k \in [1, N]$, and $k \neq i$. In Equation (6), FL denotes the set value of the identity of the individual bird playing the role of a beggar, and $FL \in (0,2]$, while setting the frequency of the swarm flying to other places as $rand$.

3.2 Convolutional Neural Network

A convolutional neural network is a feedforward neural network that includes convolutional computation and has a deep structure. It is a multilayer neural network that consists of an input layer, convolutional layer, pooling layer, fully connected layer and output layer. In a convolutional neural network, the neurons in the latter layer take the local results of the previous layer as input so that some basic feature information can be obtained. These underlying features are used by the neurons in the higher layers, and the weights of each basic feature extractor are shared globally. When the original image is input to the network, the convolutional layer is operated to obtain the feature map of the convolutional layer, the sampling layer reduces the resolution of the feature image by local averaging and sampling operations, and the semantic information is obtained by the convolutional pooling layer.

(1) Convolution layer

The convolution layer is the most important structure in the convolutional neural network, mainly for feature extraction of the original data input to the network, and multiple convolutional kernels are generally set in the convolutional layer, while the convolutional kernels are window sliding on the original image to provide more adequate features. In the neural network, discrete convolution is used, as shown in Equation (7):

$$r(m,n)=\sum_{i=0}^{M}\sum_{j=0}^{N}f(i,j)k(m-i,n-j), \tag{7}$$

where $r(m,n)$ denotes the new sequence of the convolution operation, $f(m,n)$ denotes the sequence of the convolved signal, and $k(m,n)$ denotes the convolution kernel matrix. The convolution operation in the image still follows the discrete convolution formula, and the neurons in each implicit layer share the weights and offset values of the convolution kernel, so the convolution operation formula of the convolution layer is as follows:

$$x_j^m=f(\sum_{i\in M_j}w_{ij}^m\times x_i^m+b_j^m), \tag{8}$$

where x_j^m denotes the j feature maps of the output, f denotes the activation functions of the neurons in the hidden layer, M_j denotes the set of feature maps input to the convolutional layer m, w denotes the shared weight matrix of the convolutional kernel, and b denotes the bias term.

(2) Pooling layer

After the extraction of features in the convolutional layer, feature selection and filtering must be performed, which is the role of the pooling layer (Pooling layer), through the pooling operation to reduce the dimensionality of the feature map output from the convolutional layer. The role of the pooling layer is to keep the features unchanged, perform feature downscaling and prevent network model overfitting. The general expression of the output function of the pooling layer is as follows:

$$X_j^{l+1}=f(w_j^{l+1}\times down(X_j^l)+b_j^{l+1}), \tag{9}$$

where X_j^{l+1} denotes the output after the pooling operation, $down(\bullet)$ denotes the pooling type selected by the operation, X_j^l denotes the input feature mapping, and w_j^{l+1} and b_j^{l+1} denote the weights and offsets selected by the pooling operation.

(3) Fully connected layer

The fully connected layer (FC) is the last part of the convolutional neural network, which aims to remap the convolutional neural network to the original image's labeled space by a nonlinear combination of the implicit feature maps obtained through convolution and pooling operations. This is because all neuron nodes in the fully connected layer need to be connected with all the nodes in the previous layer so that the parameters in the layer are redundant to any of the previous layers, thus producing the phenomenon of network overfitting, which ensures that the fully connected layer can integrate the features obtained from the previous convolutional and pooling layers. Then, the learning process can be freely adjusted according to the damage function to achieve the purpose of classifying the target to complete the prediction.

4 Build A Convolutional Neural Network Based on An Improved Bird Swarm Algorithm

4.1 Improved Bird Swarm Algorithm

Similar to most metaheuristic algorithms, the bird swarm algorithm also has the tendency to fall into a local optimum, leading to premature convergence of the algorithm. If the bird swarm algorithm is directly used to optimize the parameters of convolutional neural networks, this cannot bring out the performance of convolutional neural networks, resulting in better results. Therefore, in this paper, the bird swarm algorithm is optimized in two aspects to improve the performance of the algorithm, and the following improvements are made to the bird swarm algorithm.

(1) Center of gravity-based reverse population initialization

Initialization is very important for every metaheuristic algorithm, and it directly affects the quality of the solution to some extent. This is because the quality of the solution in the population comes from the qualification of both the search range and the starting position. When the initialized search range is set in a small range, it will limit the whole algorithm's ability to find the best result, while when the initial position of the algorithm is located near the global optimal solution, it can ensure that the population finds more effective individual position information in a more high-quality solution space. The bird swarm algorithm uses the ordinary random method for population initialization, which cannot satisfy the above two requirements. To solve this problem, some scholars have used backward learning strategies for population initialization with some success, but backward learning has the disadvantage that the fitness value of the current solution is the same as the fitness value of its inverse solution when facing optimization problems containing "symmetric peaks" [39]. Therefore, to overcome this drawback, we use a discrete uniform center of gravity-based backward learning strategy for population initialization. The expression is as follows:

$$C = \sum_{i=1}^{N} x_{ij} \Big/ N, \tag{10}$$

$$\overline{x}_i = C - x_i, \tag{11}$$

where N denotes the individual of the population, C denotes the population center of gravity based on discrete uniformity, and $\overline{x}_i$ is the inverse of the center of gravity of x_i . j denotes dimensionality.

(2) Optimization of foraging behavior based on similarity and aggregation

In the bird swarm algorithm in the late optimization process, the bird individuals in the swarm will gradually gather together, resulting in a certain similarity in the position of the swarm, which causes the bird swarm algorithm to easily fall into the local optimum in the late stage. To be able to effectively identify the position of the bird swarm individuals, we construct the similarity and aggregation degree, use the aggregation degree as an indicator to judge the diversity of the population in the process of foraging, and according to the diversity of the swarm the foraging position is automatically adjusted, which can allow the swarm individuals to successfully escape from the local optimum at a later stage and avoid falling into the local optimum. We use the distance between the positions of each bird in the swarm as an indicator to represent the similarity between the individuals of the swarm, and a larger difference in distance represents a higher degree of dispersion of the swarm, leading to a higher diversity, so that the similarity of individuals will be lower. The mathematical expression of similarity is as follows:

$$S_{g,u} = 1 - (x_{g,d} - x_{u,d}) \big/ \max|ub - lb|. \tag{12}$$

In Equation. (12), $S_{g,u}$ denotes the distance between individual birds g and u , $x_{g,d}$ and $x_{u,d}$ denote the location of the g th individual bird and the u th individual bird under the d -dimensional space, respectively, while ub and lb , respectively. When the result of the $S_{g,u}$ value tends to 1, it indicates that the distance between individuals of the swarm is closer and thus the similarity is higher, while when the result of $S_{g,u}$ tends to 0, it indicates that the distance between individuals is farther and the similarity is lower, thus making the location distribution of individual birds in the search space wider.

In the optimization process of the swarming algorithm, the birds will gradually aggregate until they converge next to the optimal bird individuals, and we define the aggregation degree expression as shown in Equation (13):

$$c_i(t) = \sum_{i=1}^{N} S_{i,pbest} \Big/ N, \tag{13}$$

where N denotes the swarm size, t is the number of current iterations, i denotes the swarm individuals, and $Pbest$ denotes the optimal individuals. Therefore, when the swarm is foraging collectively, each individual bird will gradually gather under the guidance of the local optimum and the whole group optimum, which tends to make the search interval where the algorithm is located gradually shrink, and once an individual bird finds the optimal food, this will result in a stagnant state; therefore, we optimize the foraging position by setting the aggregation threshold. Setting the threshold value as, the expression of the location corresponding to foraging is shown in Equation (14):

$$x_{i,j}^{t+1} = \begin{cases} x_{i,j}^{t} + (p_{i,j} - x_{i,j}^{t}) \times C \times rand \\ \quad + (g_j - x_{i,j}^{t}) \times S \times rand \;\; if(c_i(t)/c_i(1)) < \gamma \\ x_{i,j}^{t} + (p_{i,j} - x_{i,j}^{t}) \times C \times rand \\ \quad - (g_j - x_{i,j}^{t}) \times S \times rand \;\; if(c_i(t)/c_i(1)) \geq \gamma \end{cases}, \tag{14}$$

4.2 Improved Bird Swarm Algorithm for Convolutional Neural Networks

Since the initial weights in convolutional neural networks are not sensitive enough, different parameters also lead to different performances of neural networks, which may lead to network oscillation without convergence or slow convergence, thus making the training time too long. Therefore, optimizing the weights becomes a reason for improvement. Since the metaheuristic optimization algorithm does not require gradient information, the number of parameters to be adjusted is small and it has a fast convergence speed, we propose a strategy based on the improved bird swarm algorithm for convolutional neural networks (IBSA-CNN), whose core idea is that when using the IBSA algorithm to train and adjust the weights of convolutional neural networks, the weights waiting to be trained need to be vector coded in a way that starts from the characteristics of the structured analysis parameters, input to the convolutional and pooling layers of the model, as shown in Figure 1 (assuming 4 convolutional layers and 2 pooling layers). In the vector coding approach, each individual bird swarm is viewed as a vector, and each vector represents a set of weights to be trained and adjusted by the convolutional neural network in the following way:

$$\begin{aligned} Bird(i) = [&\omega_{11}^{c1}, \omega_{12}^{c1}, \omega_{13}^{c1}, \omega_{14}^{c1}, \omega_{21}^{c2}, ..., \omega_{KL}^{cn}, \\ &...\omega_{11}^{p1}, \omega_{21}^{p1}, \omega_{32}^{p1}, ...\omega_{KL}^{pn}]. \end{aligned} \tag{15}$$

In Equation (15), i denotes each individual bird swarm, and ω_{KL}^{cn} and ω_{KL}^{pn} denote the weights of the convolution kernel connecting the K th input neuron to the Gth output neuron in the n th convolutional layer and the n th pooling layer, respectively.

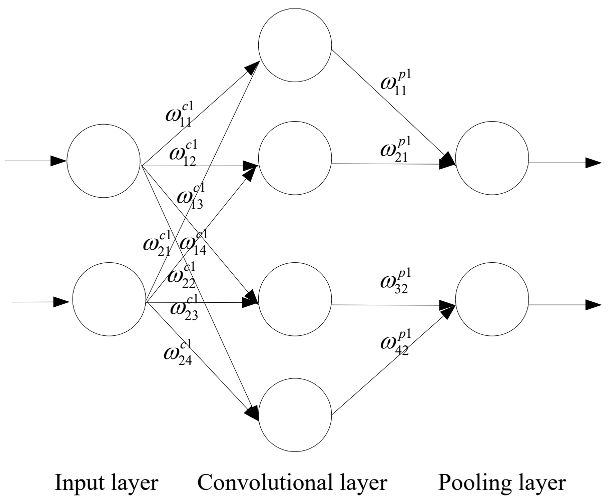


Figure 1. Convolutional layer and pooling layer weights of the convolutional neural network

We treat each bird swarm individual encoding as an N -dimensional vector, N as the dimension representing the problem waiting to be processed, and N size equal to the number of parameters of the whole convolutional neural network waiting to be trained for optimization. Finally, after determining the entire bird population size M , the encoding rule corresponding to the entire bird swarm algorithm is therefore the $M \times N$ matrix of Equation (16). In the whole matrix, ω_{KL}^{Ncn} and ω_{KL}^{Npn} denote the weights of the Ith convolutional layer and the   th pooling layer of the N th population, respectively. Therefore, we map the bird population individuals one by one into the weights and thresholds of each layer of the network after completing the encoding and matrix and calculate the mean square error generated by this neural network, as shown in (17).

$$\begin{bmatrix} \omega_{11}^{1c1},\omega_{12}^{1c1},\omega_{13}^{1c1},\omega_{14}^{1c1},\omega_{21}^{1c2},\ldots, \\ \quad \omega_{KL}^{1cn},\ldots\omega_{11}^{1p1},\omega_{21}^{1p1},\omega_{32}^{1p1},\ldots\omega_{KL}^{1pn} \\ \omega_{11}^{2c1},\omega_{12}^{2c1},\omega_{13}^{2c1},\omega_{14}^{2c1},\omega_{21}^{2c2},\ldots, \\ \quad \omega_{KL}^{2cn},\ldots\omega_{11}^{2p1},\omega_{21}^{2p1},\omega_{32}^{2p1},\ldots\omega_{KL}^{2pn} \\ \ldots. \\ \omega_{11}^{Mc1},\omega_{12}^{Mc1},\omega_{13}^{Mc1},\omega_{14}^{Mc1},\omega_{21}^{Mc2},\ldots, \\ \quad \omega_{KL}^{Mcn},\ldots\omega_{11}^{Mp1},\omega_{21}^{Mp1},\omega_{32}^{Mp1},\ldots\omega_{KL}^{Mpn} \end{bmatrix}. \tag{16}$$

$$E = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{H}(Y_{ji} - y_{ji})^2. \tag{17}$$

In Equation (17), E indicates the expected value, N indicates the number of input samples waiting to be recognized, H indicates the number of neurons output by the output layer, Y_{ji} denotes the expected output of the j th output node of the i th sample, and y_{ji} denotes the actual output of the j th output node of the i th sample. We use Equation (17) as the fitness function of IBSA, use the individual who finds the optimal weight as the weight of the convolutional neural network, and stop the calculation when the specified number of iterations or requirements are reached so that the optimal weight in the convolutional neural network can be found. The specific process is as follows.

4.3 Algorithm Steps

Step 1: Set the CNN general parameters, set the initial parameters of the bird swarm algorithm, set the number of iterations, and the population size.

Step 2: Adopt the reverse center of gravity strategy for population initialization of the bird swarm; Step 3: Adopt foraging behavior based on similarity and aggregation.

Step 4: Perform vigilance behavior and flight behavior.

Step 5: Obtain the optimal value of the individual bird swarm algorithm.

Step 6: Mapping the bird swarm individuals to the corresponding weights in the convolution and pooling layers through the correspondence principle and assigning them to the vectors in each layer by decoding.

Step 7: According to Equation (17), as the fitness function of bird swarm individuals, the calculated mean squared error is ranked by individuals.

Step 8: When the number of iterations meets the maximum number of iterations, the algorithm stops iterating and exits the algorithm loop, and the optimal solution obtained by the algorithm at this time is used as the weight parameter of the convolutional neural network; otherwise, it returns to step 3 to continue the execution.

5. Simulation Experiments

5.1 Algorithm Performance Test

Good algorithm performance is an important component to ensure the optimization of CNN model parameters and is also the key to vehicle feature recognition. We selected the Ant Colony Algorithm (ACO), Particle Swarm Optimization Algorithm (PSO), and BSA algorithm as comparison algorithms for testing, setting the number of iterations of the algorithm to 1,000 and the population size to 100. The parameters of the three algorithms were taken from the respective literature. Table 1 shows the six classical benchmark test functions, Table 2 shows the comparison of the four algorithms under different dimensions in the classical test functions, and we chose the maximum value, minimum value and variance as metrics for the comparison.

Table 1. Test function

No	Function	Test function
F1	Sphere	$f(x) = \sum_{i=1}^{n} x_i^2$
F2	Schwefel2.22	$f(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$
F3	Schwefel1.2	$f(x) = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)$
F4	Schewfel2.21	$f(x) = \max(abs(x))$

F5	Rosenbrock	$f(x)=\sum_{i=1}^{n-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$
F6	Step	$f(x)=\sum_{i=1}^{n}([x_i+0.5])^2$

Table 2. Optimization results of the four algorithms on different benchmark functions

Function	Dimension	Algorithm	Minimum value	Maximum value	Standard deviation
F1	2	ACO	0.1298	6.1872	1.5328
		PSO	0.1123	7.2918	1.1982
		BSA	1.4521E-08	2.4214E-07	4.4912E-09
		IBSA	1.9832E-13	1.1368E-10	1.7214E-16
	5	ACO	0.3706	3.9823	2.1672
		PSO	0.2873	5.1912	4.3251
		BSA	1.2531E-07	5.1361E-06	1.2831E-11
		IBSA	0	4.2316E-08	1.3138E-14
	30	ACO	124.2624	341.4163	173.8124
		PSO	132.3613	287.1238	211.7132
		BSA	2.7392E+01	1.2412E+03	2.341E+00
		IBSA	4.1733E+00	1.7634E+01	2.7123E+01
F2	2	ACO	1.2127	9.1435	5.3272
		PSO	0.0001	0.0013	0.0009
		BSA	1.9241E-06	6.1017E-03	9.3171E-04
		IBSA	0	5.1872E-22	1.5121E-32
	5	ACO	0.8474	4.1208	3.3816
		PSO	0.0049	1.3204	0.2537
		BSA	1.3261E-04	1.9112E-01	3.9013E-02
		IBSA	1.2381E-08	6.7831E-06	1.1231E-06
	30	ACO	4.4566E+02	1.7097E+05	2.4159E+04
		PSO	14.0176	27.9770	20.0343
		BSA	1.2481E-02	8.9183E+00	3.1829E+00
		IBSA	3.1491E-10	2.5221E-13	5.7191E-11
F3	2	ACO	0.0322	1.4221	0.8491
		PSO	4.0870E-11	3.0017E-08	3.1023E-09
		BSA	6.4501E-13	2.2132E-08	3.8131E-05
		IBSA	0	3.8132E-10	7.1923E-11
	5	ACO	10.2218	101.4431	27.8113
		PSO	0.0021	0.0114	0.0063
		BSA	8.4133E-07	3.3813E-05	5.1433E-03
		IBSA	1.2217E-12	5.1315E-09	1.2255E-08
	30	ACO	1.1603	9.0237	4.9412
		PSO	8.9233E-11	4.1614E-09	6.1345E-11
		BSA	3.7172E-06	9.5131E-02	1.9132E-02
		IBSA	4.6791E-11	2.8213E-07	5.0704E-08
F4	2	ACO	1.15782	8.8934	4.4321
		PSO	2.4119E-05	1.1674E-03	1.8154E-08
		BSA	9.1741E-07	2.3513E-03	4.1307E-02
		IBSA	8.2783E-19	3.1281E-15	5.2268E-16
	5	ACO	3.9816	18.9023	28.6431
		PSO	0.0517	4.4865	2.8138
		BSA	2.9321E-02	7.4451E-01	1.4127E-01
		IBSA	0	3.0684E-02	4.8915E-03
	30	ACO	83.2133	91.7246	23.4213
		PSO	15.8924	40.1832	48.1162
		BSA	6.4503E-01	1.7073E+00	2.2413E-01
		IBSA	3.8151E-02	6.3174E-01	1.5901E-02
F5	2	ACO	31.1722	62.9212	21.5236
		PSO	4.5137	7.8013	3.1916
		BSA	7.3136E-08	2.3212E+00	3.4169E-01
		IBSA	6.2673E-11	4.5916E-04	6.8130E-05
	5	ACO	18.8482	13.9522	14.2712
		PSO	11.8191	17.3633	36.1307
		BSA	8.2972E-01	3.1381E+00	1.5091E+00
		IBSA	1.6715E-03	3.1621E+00	8.8214E-01
	30	ACO	514.0331	799.3512	745.4923
		PSO	348.4001	590.2501	634.3001
		BSA	2.9221E+01	9.8061E+02	2.6021E+02
		IBSA	2.7931E+01	2.8936E+01	4.0616E-01
F6	2	ACO	1213.0169	1648.3712	4451.1354
		PSO	1512.0236	1732.7834	1871.8934
		BSA	5.7041E-10	9.9261E-04	1.9512E-04
		IBSA	1.4129E-14	6.1279E-08	1.2117E-08
	5	ACO	0.8832	0.9168	0.7932
		PSO	0.7004	0.6665	0.1309
		BSA	8.9971E-04	2.9293E-01	4.7751E-02
		IBSA	1.3212E-07	2.6103E-02	4.9214E-02
	30	ACO	2.2163	9.1312	5.2109
		PSO	1414.9936	7125.0413	1313.3237
		BSA	1.7138E+00	1.3214E+01	2.132E+00
		IBSA	3.1926E-01	2.1913E+00	6.2812E-01

The comparison results of the four algorithms under three metrics with six classical benchmarking functions are shown in Table 3. We find that the algorithms in this paper obtain better results under different conditions of the number of dimensions. This also shows that our improvement for the algorithms has good results. Among the six tested functions, IBSA has better results in terms of minimum, maximum and variance compared with the other three algorithms. In particular, in F1-F4, the minimum value is actually 0 when the dimension is 2 and 5, respectively, which shows that the algorithm has better solution quality. In F5-F6, although it does not obtain 0, it still obtains good results in terms of minimum, maximum and variance. Through these test results, we found that the performance of the algorithm was significantly improved after the center of gravity reversal strategy for population initialization and the optimization of foraging behavior based on similarity and aggregation. This lays the foundation for the subsequent development of optimization of CNN parameters to improve the accuracy of CNN recognition.

5.2 Model Performance Testing

To better illustrate the performance of IBSA-CNN, we chose the standard UCI dataset [40] to complete a simulation experiment in which we chose the UCI wine quality evaluation dataset [41]. This dataset collected 12 parameters: fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, and pH. The experimental platform was used the Linux Ubuntu operating system with the Caffe framework, Core I5 CPU, 16GDDR memory and 1T hard disk.

We set the number of layers of the convolutional neural network to six, with one input and fully connected output layer, two convolutional layers and two pooling layers. Twelve different types of wine parameters were input into the neural network, and the output was coded using a four-bit binary code to reflect the wine quality level. The results are shown in Table 3.

Table 3. Comparison of some test data

Result output				CNN				IBSA-CNN			
1	0	0	1	0.816	0.000	0.000	0.837	0.912	0.000	0.000	1.000
1	1	1	0	1.000	1.000	1.000	0.023	1.000	1.000	0.693	0.000
0	1	1	1	0.024	1.000	1.000	1.000	0.000	0.987	1.000	1.000
1	0	0	1	1.000	0.028	0.003	1.000	1.000	0.000	0.112	0.967
1	1	0	0	1.000	0.879	0.001	0.000	1.000	1.000	0.000	0.000
0	0	0	1	0.000	0.000	0.001	0.998	0.000	0.014	0.000	1.000
1	0	1	0	0.819	0.000	1.000	0.001	0.987	0.000	1.000	0.000

We used the traditional CNN model and the IBSA-CNN model in our experiments, and from the results in Table 1, the correct rate of the network evaluation of CNN reached 57.14%, while the correct rate of IBSA-CNN reached 75%. To evaluate the effect of measuring the correct rate, we modified the results of this experiment based on the existing evaluation criteria, and the evaluation formula is as follows:

$$|Y^1-y^1|+|Y^2-y^2|+|Y^3-y^3|+|Y^4-y^4|\le T. \quad \textbf{(18)}$$

We use Y^N in Eq. (18) to denote the number of binary bits expected to be output in the network and y^n to denote the actual computed output by training at bit n. T denotes the error tolerance value, and we determine the result when the error value between the two is less than the error tolerance value. The closer E is to 0, the more accurate the result is.

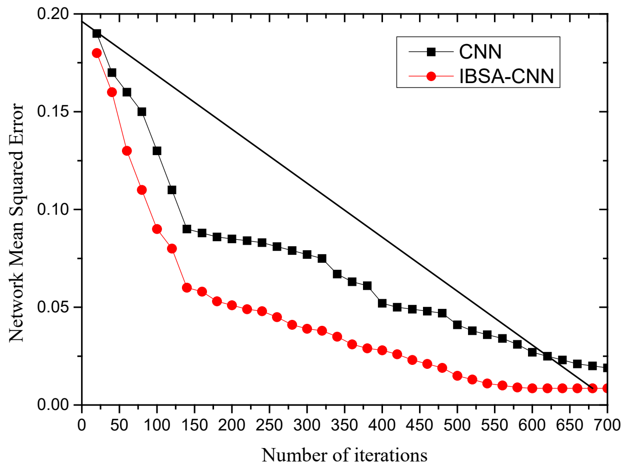


Figure 2. Comparison of the mean square deviation of the two algorithms

Figure 2 shows the comparison results of the mean squared deviation of CNN and IBSA-CNN, from which is found that both algorithms show a gradually decreasing trend at the same number of iterations, but IBSA-CNN clearly outperforms CNN, especially in the same number of iterations. IBSA-CNN outperforms CNN in both, which shows that the convergence speed of IBSA-CNN is better compared to CNN and the network handles the problem more accurately. Therefore, it shows that IBSA-CNN has more obvious advantages.

5.3 Vehicle Feature Recognition

To illustrate the effectiveness of the algorithms in this paper, we use the vehicle images from the BIT-Vehicle dataset provided in the literature [42] as the image library, which contains a total of six types of vehicles: sedan, SUV, truck, bus, microbus and minivan. The comparison algorithms we choose are CNN and R-CNN. Figure 3 shows some of the vehicle images in these data. We divide the 9,850 images in this database into two parts and allocate the training dataset and test training dataset according to 6:4. Among these, 5,910 vehicle sample images are used as training datasets, which are used to train the models of the three neural networks. The remaining 3,490 vehicle images are used as test datasets to test the comparison of the effects of the three network models completed by the network training.

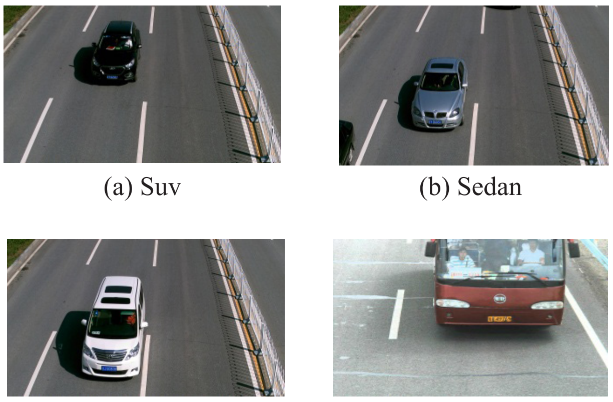


Figure 3. Some images in the database

We calculate the recall rate (R), accuracy rate (P), average precision (AP) value for a single target and mean average precision (mAP). We choose the performance metrics AP and mAP for the analysis basis. The expressions of several metrics are as follows:

$$R = \frac{T_p}{T_p + F_n}. \tag{19}$$

$$P = \frac{T_p}{T_p + F_p}. \tag{20}$$

$$mAP = \frac{\sum_{i=1}^{6} AP_i}{6}. \tag{21}$$

$$AP = \frac{\sum_{re(0,0.1,0.2,...1)} P_{inter}(r)}{11}. \tag{22}$$

$$P_{inter}(r) = \max P(r'). \tag{23}$$

In Equations (19) and (20), T_p indicates the number of positive samples correctly identified, F_p indicates the number of negative samples misclassified as positive, F_n indicates the number of positive samples misclassified as negative, and n indicates the number of all samples labeled as positive. Equation (21) expresses the value of mAP. Equation (22), $re(0, 0.1, 0.2, ..., 1)$, denotes the 11 threshold points set, so when the recall R is greater than these 11 threshold points, each recall corresponds to a maximum accuracy P, that is, Equation (23), so according to the average of these 11 points, accuracy P is the value of AP.

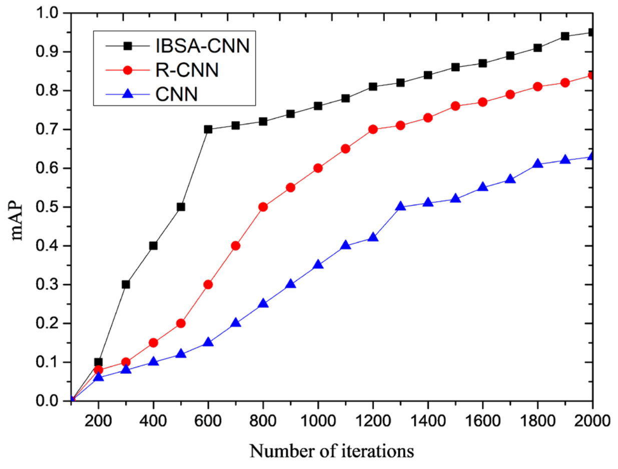


Figure 4. Comparison of the mAP of the three algorithms

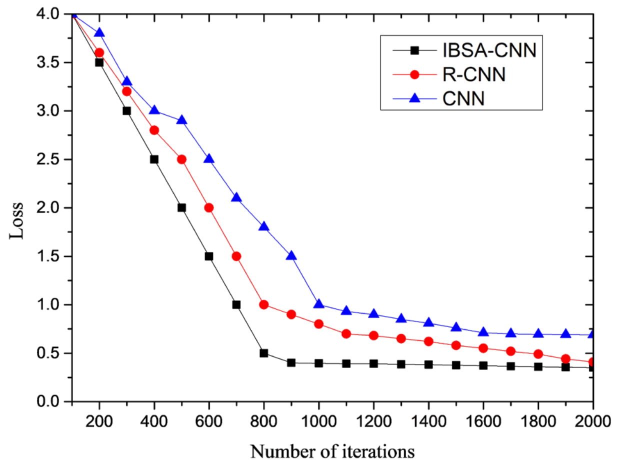
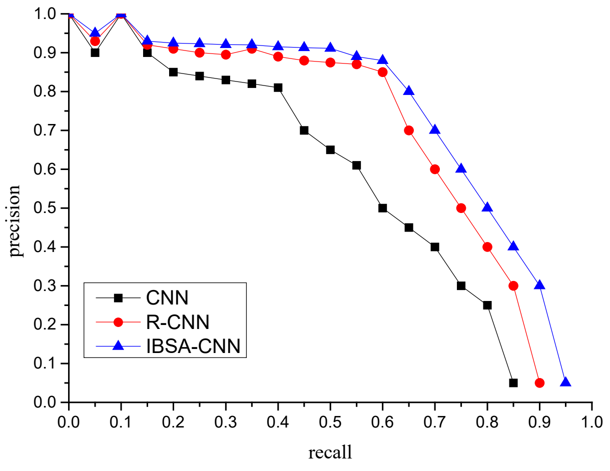


Figure 5. Comparison of the loss functions of the three algorithms

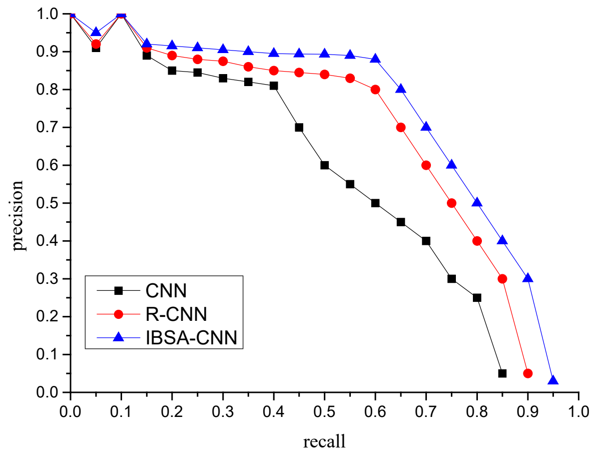
We use the produced vehicle dataset to train the R-CNN, CNN and IBSA-CNN in this study. Figure 4 shows the mAP effect of the three models for vehicle recognition The figure shows that as the number of iterations gradually increases, the results of the mAP of the three models increase to different degrees. The accuracy of the IBSA-CNN compared to the CNN and R-CNN is significant, which shows that the IBSA-CNN has better results in vehicle recognition. Figure 5 shows the results of the three models for the vehicle loss function, which shows that the IBSA-CNN has the least loss and the model has a faster learning ability and is the first to reach model convergence, while the CNN model has a slower convergence rate and a higher loss. Table 4 shows the recognition results of different neural network models for each of the six vehicle categories, illustrating that IBSA-CNN improves 4.9% compared to R-CNN and 6.8% compared to CNN in terms of recognition results in the data. Figure 6 shows the Precision-Recall curves of six kinds of car feature recognition. The results from six different car models show that the Precision values of all three algorithms exhibit a gradual decrease as the Recall values gradually increase, but IBSA-CNN has better results compared with CNN and R-CNN, which shows that IBSA-CNN has better recognition results.

Table 4. Recognition effects of different convolutional neural network models

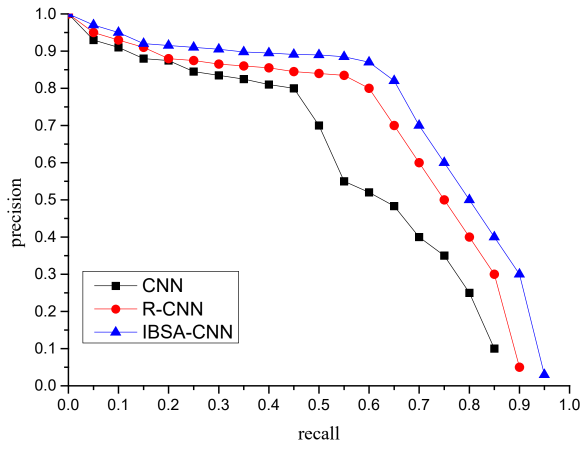
Network structure	Sedan	Suv	Truck	Bus	Microbus	Minivan	mAP
CNN	0.811	0.829	0.831	0.832	0.812	0.808	0.821
R-CNN	0.83	0.841	0.843	0.861	0.833	0.81	0.836
IBSA-CNN	0.87	0.882	0.872	0.892	0.863	0.882	0.877



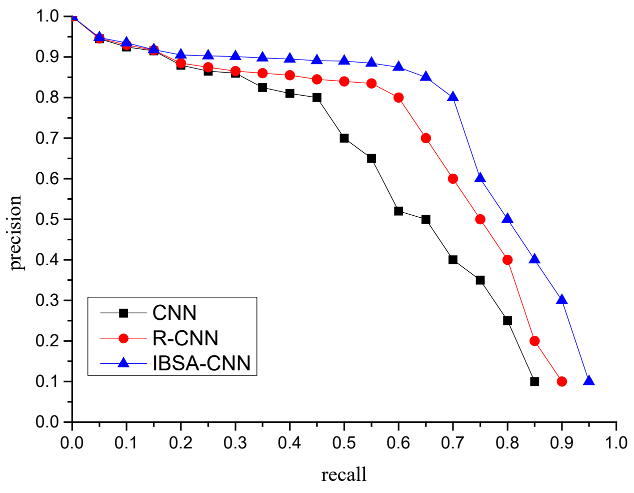
(a) Sedan



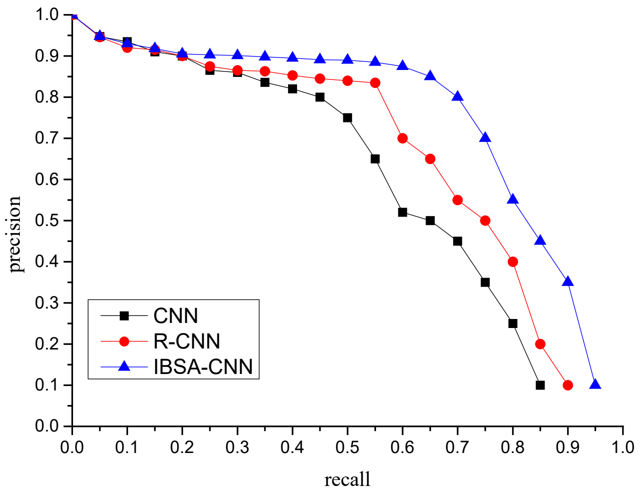
(b) Suv



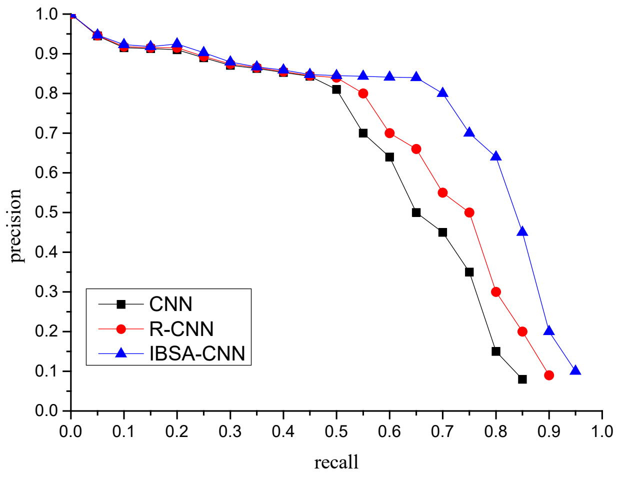
(c) Truck



(d) Bus



(e) Minivan



(f) Microbus

Figure 6. P-R curve value

6. Conclusion

To address the problems of local optima, slow convergence and weak generalization ability of CNNs, we propose an improved strategy for the bird swarm algorithm to optimize the weights of convolutional and pooling layers in convolutional neural networks. We use a center of gravity backward learning strategy in population initialization to improve individual diversity and an optimization strategy based on similarity and aggregation in foraging behavior to prevent the bird swarm algorithm from falling into a local optimum, which improves the algorithm performance. In the simulation experiments, we compare CNN and R-CNN in the BIT-Vehicle dataset. The test results illustrate that IBSA-CNN has better vehicle feature recognition.

Acknowledgements

This work is supported by Zhejiang Provincial Philosophy and Social Sciences Planning Project (23NDJC369YB).

References

[1] J. P. Zhang, F. Y. Wang, K. F. Wang, W. H. Lin, X. Xin, C. Chen, Data-driven intelligent transportation systems: A survey, *IEEE Transactions on Intelligent Transportation Systems*, Vol. 12, No. 4, pp. 1624-1639, December, 2011.

[2] B. Jia, R. Liu, M. Zhu, Real-time obstacle detection with motion features using monocular vision, *The Visual Computer*, Vol. 31, No. 3, pp. 281-293, March, 2015.

[3] J. Gu, Z. Wang, J. Kuen, L. Y. Ma, A. Shahroudy, B. Shuai, T. Liu, X. X. Wang, G. Wang, J. F. Cai, T. Chen, Recent advances in convolutional neural networks, *Pattern recognition*, Vol. 77, pp. 354-377, May, 2018.

[4] X. B. Meng, X. Z. Gao, L. Lu, Y. Liu, H. Z. Zhang, A New bio-inspired optimization algorithm: Bird Swarm Algorithm, *Journal of Experimental & Theoretical Artificial Intelligence*, Vol. 28, No. 4, pp. 673-687, 2016.

[5] K. S. Raghunandan, P. Shivakumara, L. Padmanabhan L, G. H. Kumar, T. Lu, U. Pal, Symmetry features for license plate classification, *CAAI Transactions on Intelligence Technology*, Vol. 3, No. 3, pp. 176-183, September, 2018.

[6] S. P. Adhikari, H. T. Cho, H. J. Yoo, C. J. Yang, H. S. Kim, On-road succeeding vehicle detection using characteristic visual features, *The transactions of The Korean Institute of Electrical Engineers*, Vol. 59, No. 3, pp. 636-644, March, 2010.

[7] N. Otsu, A threshold selection method from gray-level histograms, *IEEE transactions on systems, man, and cybernetics*, Vol. 9, No. 1, pp. 62-66, January, 1979.

[8] T. N. Tan, K. D. Baker, Efficient image gradient based vehicle localization, IEEE Transactions on Image Processing, Vol. 9, No. 8, pp. 1343-1356, August, 2000.

[9] G. S. K. Fung, N. H. C. Yung, G. K. H. Pang, Vehicle shape approximation from motion for visual traffic surveillance, *2001 IEEE Intelligent Transportation Systems*, Oakland, CA, USA, 2001, pp. 608-613.

[10] M. Cheon, W. Lee, C. Yoon, M. Park, Vision-based vehicle detection system with consideration of the detecting location, *IEEE transactions on intelligent transportation systems*, Vol. 13, No. 3, pp. 1243-1252,

September, 2012.

[11] Z. Sun, G. Bebis, R. Miller, Monocular precrash vehicle detection: features and classifiers, *IEEE transactions on image processing*, Vol. 15, No. 7, pp. 2019-2034, July, 2006.

[12] Z. Sun, G. Bebis, R. Miller, Quantized wavelet features and support vector machines for on-road vehicle detection, *7th International Conference on Control, Automation, Robotics and Vision*, Singapore, 2002, pp. 1641-1646.

[13] C. Caraffi, T. Vojíř, J. Trefný, J. Šochman, J. Matas, A system for real-time detection and tracking of vehicles from a single car-mounted camera, *15th international IEEE conference on intelligent transportation systems*, Anchorage, AK, USA, 2012, pp. 975-982.

[14] Y. M. Baek, W. Y. Kim, Forward vehicle detection using cluster-based AdaBoost, *Optical Engineering*, Vol. 53, No. 10, pp. 1-15, April, 2014.

[15] L. Leyrit, T. Chateau, C. Tournayre, J. Lapreste, Association of AdaBoost and kernel based machine learning methods for visual pedestrian recognition, *2008 IEEE Intelligent Vehicles Symposium*, Eindhoven, Netherlands, 2008, pp. 67-72.

[16] A. Khammari, F. Nashashibi, Y. Abramson, C. Laurgeau, Vehicle detection combining gradient analysis and AdaBoost classification, *2005 IEEE Intelligent Transportation Systems*, Vienna, Austria, 2005, pp. 1084-1089.

[17] A. Broggi, E. Cardarelli, S. Cattani, P. Medici, M. Sabbatelli, Vehicle detection for autonomous parking using a Soft-Cascade AdaBoost classifier, *2014 IEEE Intelligent Vehicles Symposium Proceedings*, Dearborn, MI, USA, 2014, pp. 912-917.

[18] Z. Sun, G. Bebis, R. Miller, On-road vehicle detection using evolutionary Gabor filter optimization, *IEEE Transactions on Intelligent Transportation Systems*, Vol. 6, No. 2, pp. 125-137, June, 2005.

[19] R. T. Collins, A. J. Lipton, T. Kanade, Introduction to the special section on video surveillance, *IEEE Transactions on pattern analysis and machine intelligence*, Vol. 22, No. 8, pp. 745-746, August, 2000.

[20] D. K. Dewangan, S. P. Sahu, RCNet: road classification convolutional neural networks for intelligent vehicle system, *Intelligent Service Robotics*, Vol. 14, No. 2, pp. 199-214, April, 2021.

[21] M. Maity, S. Banerjee, S. S. Chaudhuri, Faster R-CNN and yolo based vehicle detection: A survey, *5th International Conference on Computing Methodologies and Communication (ICCMC)*, Erode, India, 2021, pp. 1442-1447.

[22] Z. Charouh, A. Ezzouhri, M. Ghogho, Z. Guennoun, A resource-efficient CNN-based method for moving vehicle detection, *Sensors*, Vol. 22, No. 3, Article No. 1193, February, 2022.

[23] J. Luo, H. Fang, F. Shao, Y. Zhong, X. Hua, Multi-scale traffic vehicle detection based on faster R–CNN with NAS optimization and feature enrichment, *Defence Technology*, Vol. 17, No. 4, pp. 1542-1554, August, 2021.

[24] J. Hu, Y. Sun, S. Xiong, Research on the Cascade Vehicle Detection Method Based on CNN, *Electronics*, Vol. 10, No. 4, Article No. 481, February, 2021.

[25] R. Ghosh, On-road vehicle detection in varying weather conditions using faster R-CNN with several region proposal networks, *Multimedia Tools and Applications*, Vol. 80, No. 17, pp. 25985-25999, July, 2021.

[26] S. Karungaru, L. Dongyang, K. Terada, Vehicle detection and type classification based on CNN-SVM, *International Journal of Machine Learning and Computing*, Vol. 11, No. 4, pp. 304-310, July, 2021.

[27] M. Anandhalli, V. P. Baligar, P. Baligar, P. Deepsir, M. Iti, Vehicle detection and tracking for traffic management, *IAES International Journal of Artificial Intelligence*, Vol. 10, No. 1, pp. 66-73, March, 2021.

[28] F. C. Soon, H. Y. Khaw, J. H. Chuah, J. Kanesan, Hyper-parameters optimisation of deep CNN architecture for vehicle logo recognition, *IET Intelligent Transport Systems*, Vol. 12, No. 8, pp. 939-946, October, 2018.

[29] H. Y. Fu, H. D. Ma, G. Y. Wang, X. M. Zhang, Y. F. Zhang, MCFF-CNN: Multiscale comprehensive feature fusion convolutional neural network for vehicle color recognition based on residual learning, *Neurocomputing*, Vol. 395, pp. 178-187, June, 2020.

[30] S. C. Hsu, C. L. Huang, C. H. Chuang, Vehicle detection using simplified fast R-CNN, *2018 International Workshop on Advanced Image Technology (IWAIT)*, Chiang Mai, Thailand, 2018, pp. 1-3.

[31] V. Kukreja, D. Kumar, A. Kaur, Geetanjali, Sakshi, GAN-based synthetic data augmentation for increased CNN performance in Vehicle Number Plate Recognition, *2020 4th international conference on electronics, communication and aerospace technology (ICECA)*, Coimbatore, India, 2020, pp. 1190-1195.

[32] A. Tourani, S. Soroori, A. Shahbahrami, S. Khazaee, A. Akoushideh, A robust vehicle detection approach based on faster R-CNN algorithm, *2019 4th International Conference on Pattern Recognition and Image Analysis (IPRIA)*, Tehran, Iran, 2019, pp. 119-123.

[33] Y. Huang, R. Wu, Y. Sun, W. Wang, X. Ding, Vehicle logo recognition system based on convolutional neural networks with a pretraining strategy, *IEEE Transactions on Intelligent Transportation Systems*, Vol. 16, No. 4, pp. 1951-1960, August, 2015.

[34] P. Kaur, Y. Kumar, S. Ahmed, A. Alhumam, R. Singla, M. F. Ijaz, Automatic license plate recognition system for vehicles using a CNN, CMC-Computers, *Materials & Continua*, Vol. 71, No. 1, pp. 35-50, November, 2021.

[35] Y. Zhang, J. Li, Y. Guo, C. Xu, J. Bao, Y. Song, Vehicle driving behavior recognition based on multi-view convolutional neural network with joint data augmentation, *IEEE Transactions on Vehicular Technology*, Vol. 68, No. 5, pp. 4223-4234, May, 2019.

[36] S. Q. Li, W. Li, S. P. Wen, K. B. Shi, Y. Yang, P. Zhou, T. W. Huang, Auto-FERNet: A facial expression recognition network with architecture search, *IEEE Transactions on Network Science and Engineering*, Vol. 8, No. 3, pp. 2213-2222, July-September, 2021.

[37] W. Li, S. P. Wen, K. B. Shi, Y. Yang, T. W. Huang, Neural architecture search with a lightweight transformer for text-to-image synthesis, *IEEE*

Transactions on Network Science and Engineering, Vol. 9, No. 3, pp. 1567-1576, May-June, 2022.

[38] B. Lyu, S. P. Wen, K. B. Shi, T. W. Huang, Multiobjective reinforcement learning-based neural architecture search for efficient portrait parsing, *IEEE Transactions on Cybernetics*, pp. 1-12, August, 2021. DOI: 10.1109/TCYB.2021.3104866.

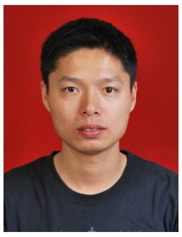
[39] Z. H. Chen, Z. H. Zhan, Two-Layer collaborative differential evolution algorithm for multimodel optimization problems, *Chinese Journal of Computers*, Vol. 44, No. 9, pp. 1806-1823, September, 2021.

[40] UCI Machine Learning Repository, Wine Quality Data Set, http://archive.ics.uci.edu/ml/datasets/Wine+Quality.

[41] P. Cortez, A. Cerdeira, F. Almeida, T. Matosb, J. Reisab, Modeling wine preferences by data mining from physicochemical properties, *Decision support systems*, Vol. 47, No. 4, pp. 547-553, November, 2009.

[42] Z. Dong, Y. Wu, M. Pei, Y. D. Jia, Vehicle type classification using a semisupervised convolutional neural network, *IEEE transactions on intelligent transportation systems*, Vol. 16, No. 4, pp. 2247-2256, August, 2015.

Biography



Xuan Chen is an associate professor at Zhejiang Industry Polytechnic College. He received his master degree from University of Electronic Science and Technology of China. His research interests include cloud computing and algorithm design.