Contents lists available at ScienceDirect

Measurement: Sensors

journal homepage: www.sciencedirect.com/journal/measurement-sensors

Cars logo recognition by using of backpropagation neural networks

Jabbar Majeed Sadeq [a,*], Brzo Aziz Qadir [b], Hayder Hassan Abbas [c]

[a] Information & Communication Technology Department, Erbil Technology College, Erbil Polytechnic University, Erbil, Iraq
[b] Automation Industrial Technology, Erbil Technology College, Erbil Polytechnic University, Erbil, Iraq
[c] Department of Chemical Engineering, Koya University, Kurdistan Region, Iraq

ARTICLE INFO

Keywords:
ANN
Back-propagation
Feed forward neural networks
Cars logo recognition

ABSTRACT

Numerous developments in creating "intelligent" programs have been made, some of which have drawn inspiration from biological neural networks. Researchers are developing artificial neural networks (ANNs) from various scientific fields to address multiple issues in prediction, control, and optimization. Many s, densely connected processors make up artificial neural networks, which can be considered parallel and distributed processing systems. This paper uses a back propagation ANN algorithm to recognize cars' logos. Sixty-eight images are used, which represents the logo of vehicles and their distortion and rotation, to simulate human eye recognition. The proposed design is trained by using a supervised learning algorithm. The designed system successfully validates the car logos by 99.6%, which consider a high percentage in such a field. Finally, the implemented system presents a real-life application of ANN.

1. Introduction

The security issue is a big problem for many countries nowadays; car Logo Recognition (CLR) is a crucial component of studying how vehicles behave and can offer more data for robotic systems to identify the car. Recent years have seen a rise in traffic monitoring information systems, which has sped up the development of vision-based vehicle recognition technology, a hotspot for robotics research. The management of traffic and public security is largely maintained by vehicle identification. In the actual world, car-related crimes and traffic accidents frequently happen. The functioning of the license plate recognition system will typically be unsteady due to a polluted or concealed vehicle license plate. Additionally, some people and criminals even create automobiles with fake license plates to escape taxes and interfere with regular transit. The car Logo Recognition (CLR) is addressed in this paper is used back-propagation Algorithm Artificial Neural Network (BP-ANN), which is used in many pattern recognition systems. The Car logo recognition system has bee addressed by many researchers in previous works. For instance, inn [1], the authors proposed a convolutional neural network (CNN) system for VMR that removes the requirement for precise logo detection and segmentation. The average accuracy of 99.07% is obtained, demonstrating the high classification potential and robustness against various poor imaging situations. The authors in Ref. [2] are using a single deep neural network based on a reduced ResNeXt model, and Feature Pyramid Networks is proposed in this paper, which is named as Single Shot Feature Pyramid Detector (SSFPD) the simulation results of Vehicle Logos Dataset and 99.52% accuracy on another public dataset are archived. Therefore, in this pape,r a back propagation artificial neural network is proposetoto build an artificial robotic system for the Car Logo Recognition system. The proposed system uses the simplest, easy to train and cheap to implement algorithms which coconsiderhe more applicable related to cost and have the feasibility to solve real-life problems (see Fig. 1).

The structure and operations of biological neural networks are attempted to be modelled mathematically using Artificial Neural Networks (ANNs) [3]. A simple mathematical model called an artificial neuron works as the core of every artificial neural network (process). Three straightforwardde laws govern this model: activation, addition, and multiplication. Synthet neuron inputs are weighted at the entrance, which implies that the weight is multiplied by each input value. All weighted inputs and biases are summed using the sum function in the artificial neuron's central region. At the exit of the artificial neuron, the bias anicd the total of previously weighted inputs are combined [4].

Although the operation basics and a set of rules of artificial neurons appear unremarkable, their full potential and measurements of power are realised when we connect them to artificial neural networks. This is due to the simple fact that complexity can be grown out of a few basic and simple rules.

* Corresponding author.
E-mail addresses: Jabbar.sadeq@epu.edu.iq (J. Majeed Sadeq), brzo.qadir@epu.edu.iq (B. Aziz Qadir), Hayder.hassan@koyauniversity.org (H. Hassan Abbas).

https://doi.org/10.1016/j.measen.2023.100702
Received 28 October 2022; Received in revised form 11 January 2023; Accepted 6 February 2023
Available online 8 February 2023

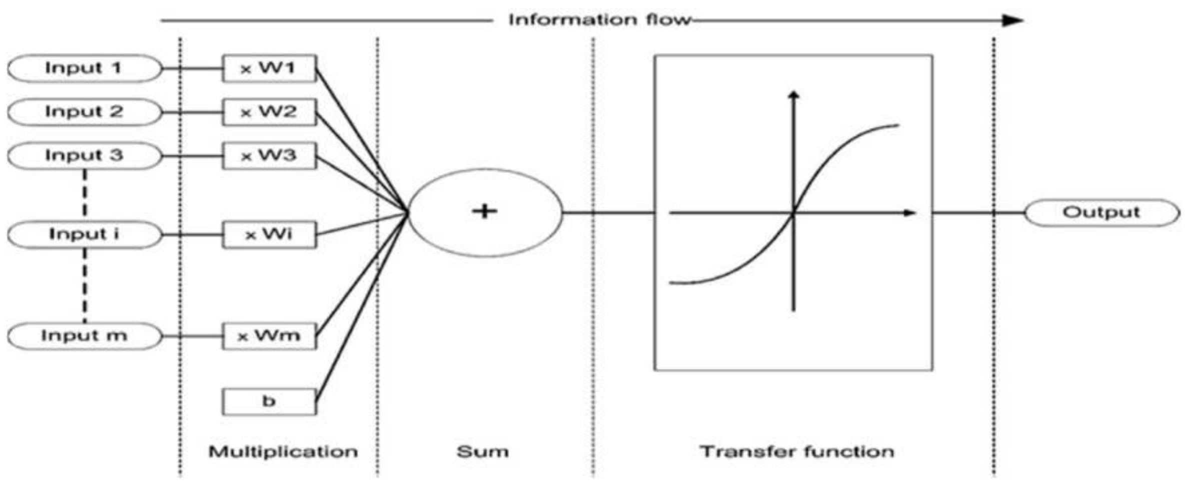


Fig. 1. Working principles of an artificial neuron [3].

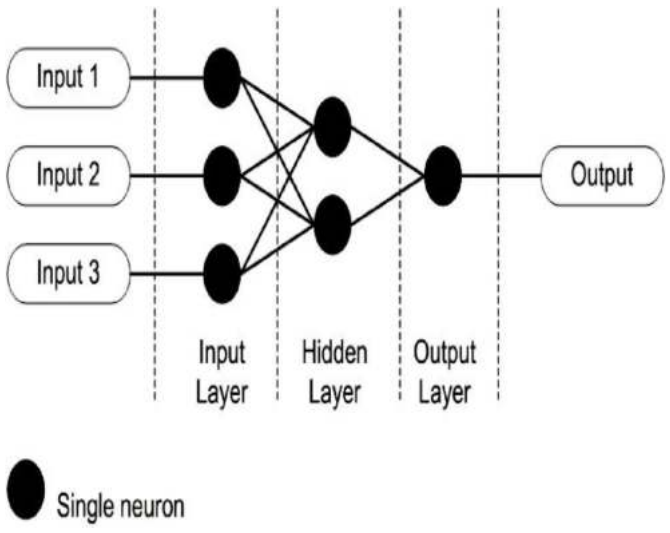


Fig. 2. Example of a simple artificial neural network.

The interconnect of these artificial neurons randomly to fully reap the benefits of mathematical complexity that can be obtained by the connection of individual neurons, rather than simply making the system complex and unmanageable. Several "standards" topographies of artificial neural networks have been proposed. Different forms of artificial neural network alalgorithmskare appropriate for tackling different types of problems, and this predetermined topology can assist us with more accessible, faster, and more efficient problem-solving. After establishing the type of problem, the most efficient architecture of the artificial neural network will be used.

2. The algorithm

The Back Propagation Network is widely regarded as the classic Neural Network. Rather than the network itself, Back Propagation1,2,3 is the training or learning method [5]. On Hopfield Networks, these are known as Feed-Forward Networks or Multi-Layer Perceptrons (MLPs) [6]. The network functions in the same manner as the others we've seen. Let's look at what Back Propagation is and how to apply it.

A Back Propagation Network (BPN) is a network that learns by doing. You provide the algorithm examples of what you want the web to do, and it adjusts the network's weights so that, once trained, it produces the desired output for a given input. Back Propagation.

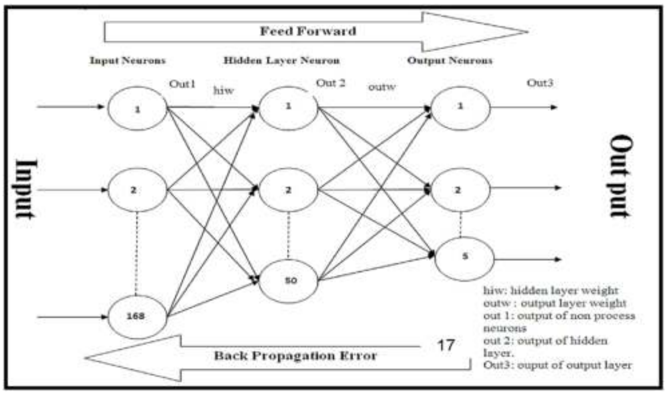


Fig. 3. The proposed BP [9].

2.1. Neural networks for backpropagation techniques

Neural networks are high-performance computing systems that process data faster than standard computing methods [7,8] (see Fig. 2).

Artificial Neural Networks, which use the backpropagation technique to recognize patterns, are an excellent and extensively used solution to pattern recognition difficulties. This project aims to create an

Fig. 4. Cars logo [1,2].

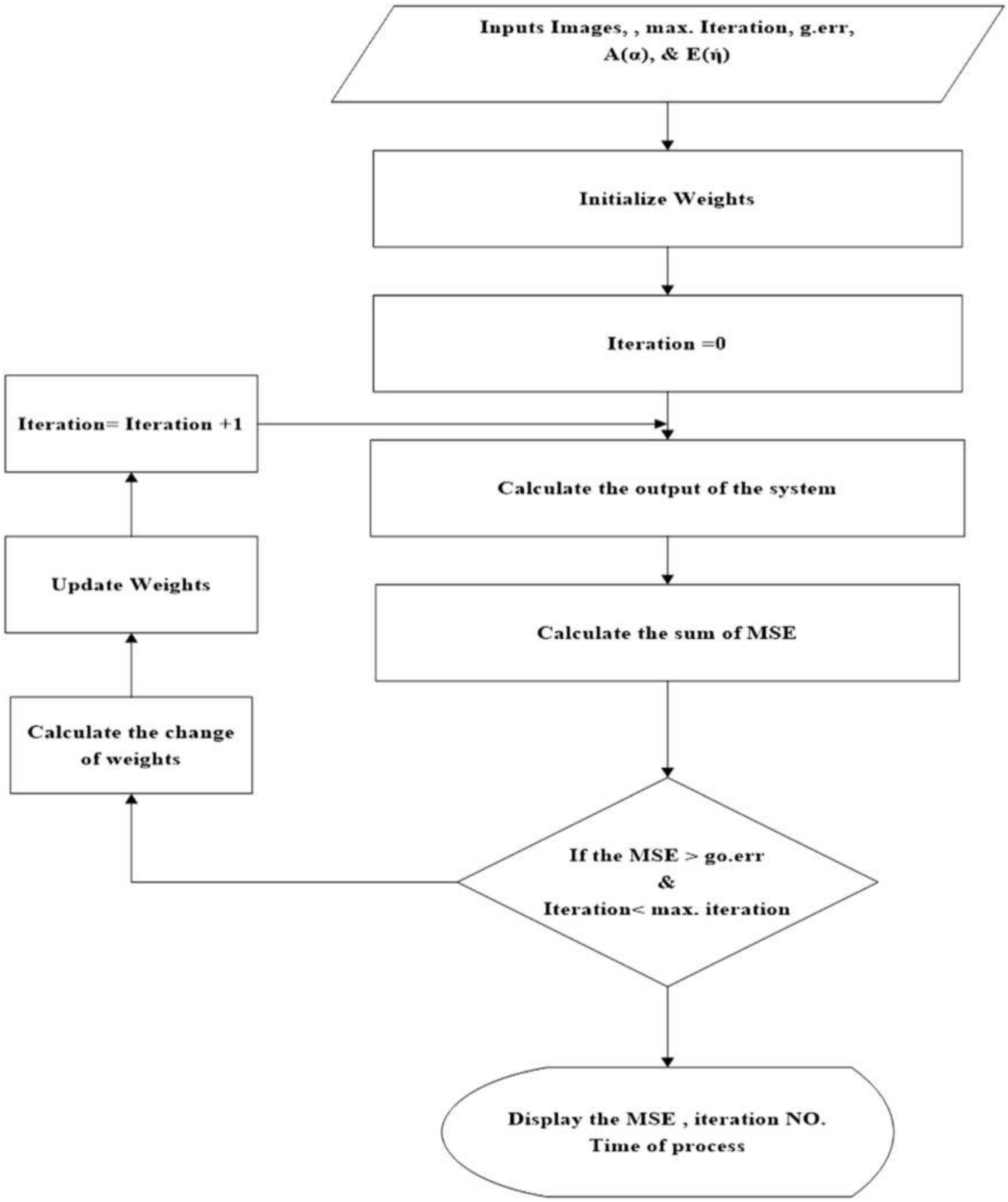

Inputs Images, , max. Iteration, g.err, A(α), & E(ή)
Initialize Weights
Iteration =0
Iteration= Iteration +1
Calculate the output of the system
Update Weights
Calculate the sum of MSE
Calculate the change of weights
If the MSE > go.err & Iteration< max. iteration
Display the MSE , iteration NO. Time of process

Fig. 5. The flowchart of software program [10].

efficient automobile logo identification system. The constructed program uses the ANN toolbox in the MATLAB package, and the database is made up of (17) car logos that are identified using a back-propagation neural network. Fig. 4 depicts the car logo used as part of the identification system; the graphics were downsized to 20*20 pixels. Fig. 3 shows the proposed analysis architecture employed in this study, whereas Table 1 lists the parameters used (see Fig. 5).

To identify the automobiles logo, the proposed networks have 400 input neurons and 17 output neurons in the output layer. The network has two layers: a log-sigmoid transfer function and 50 neurons in the hidden layer [9].

2.2. Implemented BPANN

In this section, the BPANN solves the character recognition problem using Matlab programming. The system was trained to recognize the 68 automobiles logotype using the suggested BPANN architecture, both with and without noise. This part will go over the experimental results and the programming steps.

The input database for this project consists of 68 images of automobile logos, 34 of which are noise images, which were added using paint software.

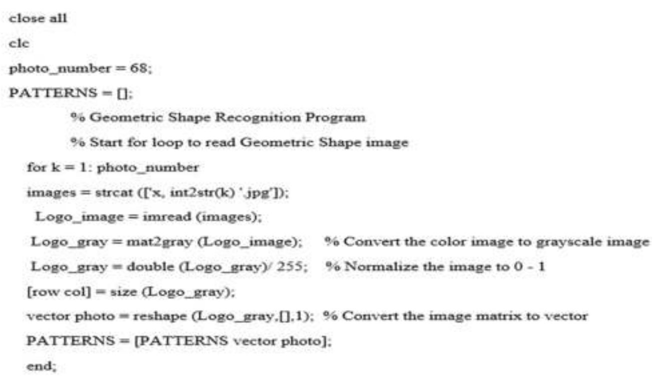
Using the Photoshop application, these photos were transformed into programming-ready visuals. This application was used to convert a color image to a Cray image and to adjust the resolution. as seen in the diagram below, Images with a resolution of 20 × 20 pixels are familiar. Fig. 4 depicts the many car logos used in the training and testing procedure.

4. Proposed system design and flow charts.

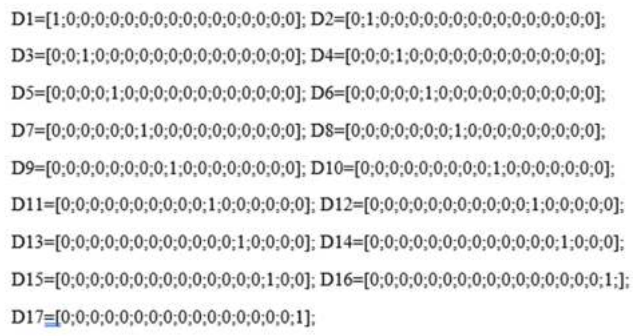
Matlab programming was used to develop the BPANN algorithm. The programs go through three phases: design, training, and testing to deploy neural networks. As seen in the diagram below.

The system was trained on four groups of 17 automobile logos (17 × 4 = 68), which were represented by matrices before being converted to grey codes. Then Matlab software was used to teach them.

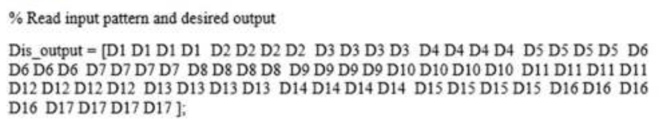
First, use the image read function in MATLAB to read the input photos and create a loop for training and testing as follows:



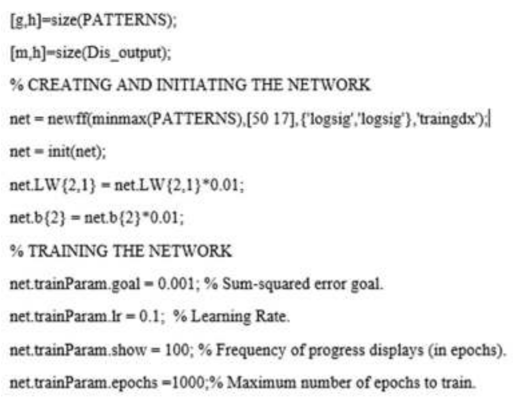
```
close all
clc
photo_number = 68;
PATTERNS = [];
           % Geometric Shape Recognition Program
           % Start for loop to read Geometric Shape image
   for k = 1: photo_number
   images = strcat (['x, int2str(k) '.jpg']);
    Logo_image = imread (images);
   Logo_gray = mat2gray (Logo_image);     % Convert the color image to grayscale image
   Logo_gray = double (Logo_gray)/ 255;   % Normalize the image to 0 - 1
   [row col] = size (Logo_gray);
   vector photo = reshape (Logo_gray,[],1);  % Convert the image matrix to vector
   PATTERNS = [PATTERNS vector photo];
   end;
```



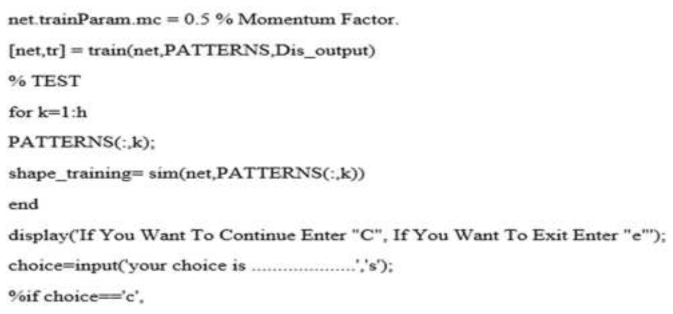
```
D1=[1;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0]; D2=[0;1;0;0;0;0;0;0;0;0;0;0;0;0;0;0];
D3=[0;0;1;0;0;0;0;0;0;0;0;0;0;0;0;0]; D4=[0;0;0;1;0;0;0;0;0;0;0;0;0;0;0;0];
D5=[0;0;0;0;1;0;0;0;0;0;0;0;0;0;0;0]; D6=[0;0;0;0;0;1;0;0;0;0;0;0;0;0;0;0];
D7=[0;0;0;0;0;0;1;0;0;0;0;0;0;0;0;0]; D8=[0;0;0;0;0;0;0;1;0;0;0;0;0;0;0;0];
D9=[0;0;0;0;0;0;0;0;1;0;0;0;0;0;0;0]; D10=[0;0;0;0;0;0;0;0;0;1;0;0;0;0;0;0];
D11=[0;0;0;0;0;0;0;0;0;0;1;0;0;0;0;0]; D12=[0;0;0;0;0;0;0;0;0;0;0;1;0;0;0;0];
D13=[0;0;0;0;0;0;0;0;0;0;0;0;1;0;0;0]; D14=[0;0;0;0;0;0;0;0;0;0;0;0;0;1;0;0];
D15=[0;0;0;0;0;0;0;0;0;0;0;0;0;0;1;0;0]; D16=[0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;1;];
D17=[0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;1];
```



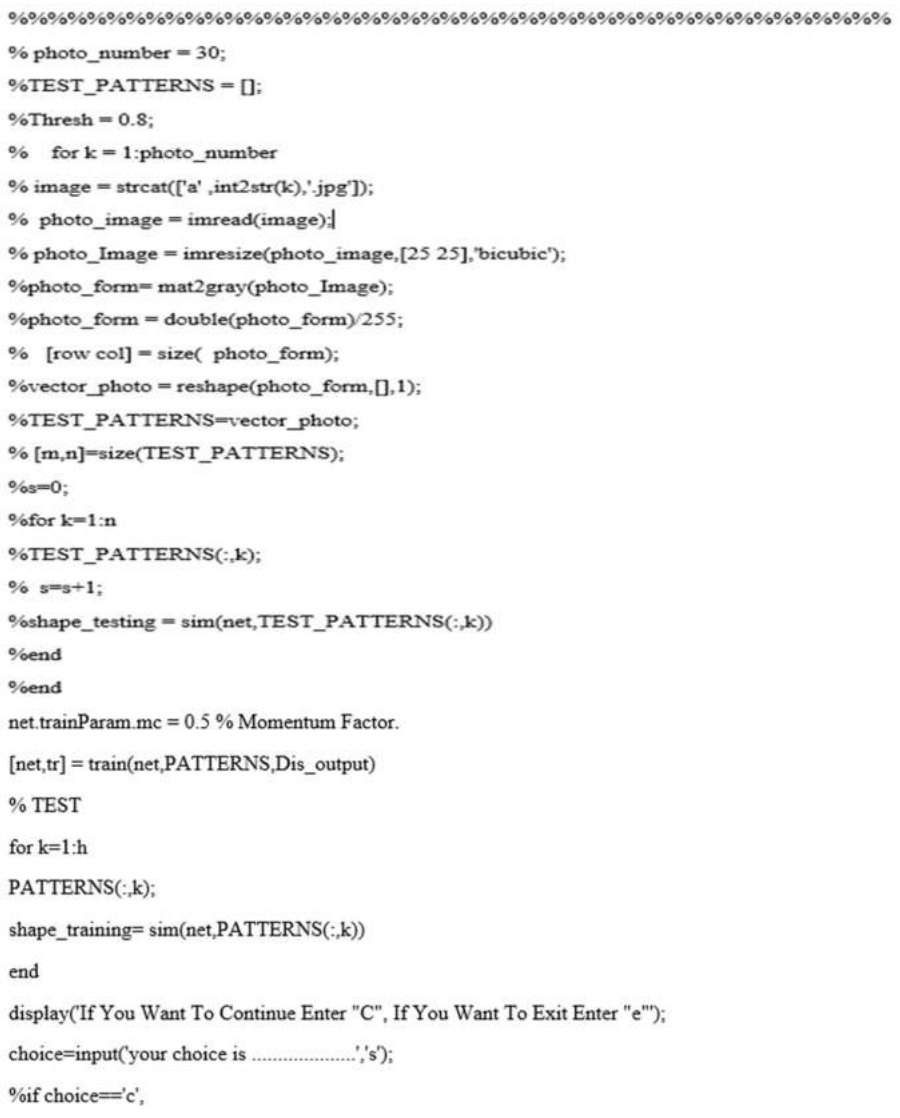
```
% Read input pattern and desired output
Dis_output = [D1 D1 D1 D1  D2 D2 D2 D2  D3 D3 D3 D3  D4 D4 D4 D4  D5 D5 D5 D5  D6
D6 D6 D6  D7 D7 D7 D7  D8 D8 D8 D8  D9 D9 D9 D9 D10 D10 D10 D10  D11 D11 D11 D11
D12 D12 D12 D12  D13 D13 D13 D13  D14 D14 D14 D14  D15 D15 D15 D15  D16 D16  D16
D16  D17 D17 D17 D17 ];
```



```
[g,h]=size(PATTERNS);
[m,h]=size(Dis_output);
% CREATING AND INITIATING THE NETWORK
net = newff(minmax(PATTERNS),[50 17],{'logsig','logsig'},'traingdx');
net = init(net);
net.LW{2,1} = net.LW{2,1}*0.01;
net.b{2} = net.b{2}*0.01;
% TRAINING THE NETWORK
net.trainParam.goal = 0.001; % Sum-squared error goal.
net.trainParam.lr = 0.1;  % Learning Rate.
net.trainParam.show = 100; % Frequency of progress displays (in epochs).
net.trainParam.epochs =1000;% Maximum number of epochs to train.
```



```
net.trainParam.mc = 0.5 % Momentum Factor.
[net,tr] = train(net,PATTERNS,Dis_output)
% TEST
for k=1:h
PATTERNS(:,k);
shape_training= sim(net,PATTERNS(:,k))
end
display('If You Want To Continue Enter "C", If You Want To Exit Enter "e"');
choice=input('your choice is ...................','s');
%if choice=='c',
```



```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% photo_number = 30;
%TEST_PATTERNS = [];
%Thresh = 0.8;
%    for k = 1:photo_number
% image = strcat(['a' ,int2str(k),'.jpg']);
%  photo_image = imread(image);
% photo_Image = imresize(photo_image,[25 25],'bicubic');
%photo_form= mat2gray(photo_Image);
%photo_form = double(photo_form)/255;
%   [row col] = size(  photo_form);
%vector_photo = reshape(photo_form,[],1);
%TEST_PATTERNS=vector_photo;
% [m,n]=size(TEST_PATTERNS);
%s=0;
%for k=1:n
%TEST_PATTERNS(:,k);
%  s=s+1;
%shape_testing = sim(net,TEST_PATTERNS(:,k))
%end
%end
net.trainParam.mc = 0.5 % Momentum Factor.
[net,tr] = train(net,PATTERNS,Dis_output)
% TEST
for k=1:h
PATTERNS(:,k);
shape_training= sim(net,PATTERNS(:,k))
end
display('If You Want To Continue Enter "C", If You Want To Exit Enter "e"');
choice=input('your choice is ....................','s');
%if choice=='c',
```

. (*continued*).

Table 1
Parameter used in the proposed BPANN.

Max. iteration	1000
ή (learning coefficient)	0.1
α(momentum factor)	0.5
Hidden layer weight	-1–1
Output layer weight	-1–1
No. of hidden layer neuron.	50
Bias	0
Sum-squared error goal	0.01

Table 2
Training parameters after running in MATLAB.

Parameters	Values
Image number of training	68
Number of epochs	312 iterations
Number of maximum iteration	1000
Performance	0.000994
Processing time	0.00:06
Gradient	0.000186
Validation checks	0

Initialization of weights: The second step is to initialize the weights. The program adjusts the values of hiding and out, which are the hidden and output layer weights, respectively, in this section. The weight of the concealed layer is then changed, and the output layer is set to zero. After running in Matlab, training checking is shown in Fig. 6. The following parameters are presented in Table 2.

Feeding forward is the third phase.

The program calculates the output from the two layers, calculates the system's Mean square error (MSE), and compares the result to the goal error in this part. If the MSE error is larger than the target error and the number of iterations is fewer than the maximum iteration, the process will be repeated until one of the conditions is met.

Fourth step: Error backpropagation: Calculate the derivative of the log-sigmoid function in this stage of the algorithm, and use these results to update the weight.

Calculate the derivative of the log-sigmoid function in this stage of the algorithm, and use these results to update the weight.

5. Updating the weights is the fifth stage. In this part, the weights have been modified to reduce the discrepancy between the output and the desired output.
6. Finally, when the program achieves the objective error, the error, the number of iterations, and the processing time are displayed.

The application is designed to evaluate two types of data: the first is test input photographs of cars' logos, and the second is noise images.

2.3. Results and discussion

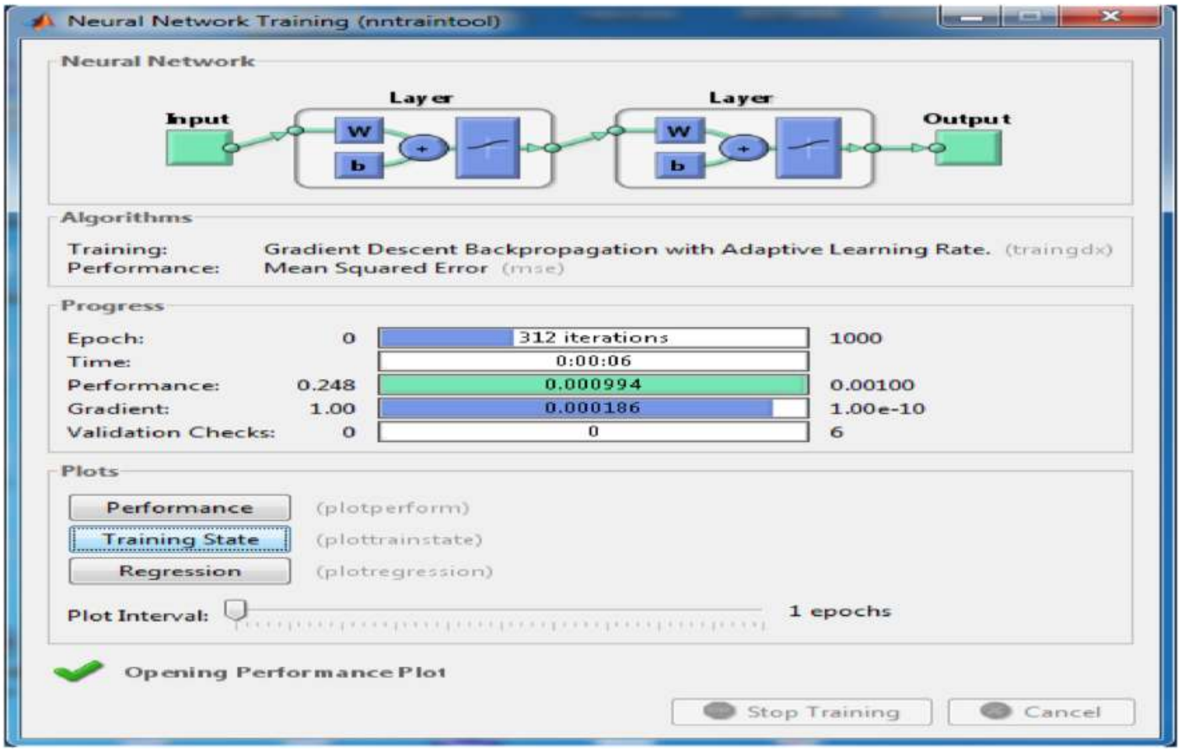
The proposed BPANN algorithm and Matlab package version 2010a, a good version of Matlab, are used to implement the program's results. The ANN parameters have been fixed and selected based on numerous test processes, and these values have been considered the best in our instance. The proper identification rate for varied training and testing data is calculated by the number of correct later divided by the total tested later. The values of 0.01, 0.45, and 50 for the number of neurons are correspondingly 0.01, 0.45, and 50.

The mean square error vs iteration is shown in Fig. 7.

3. Conclusion

BPANN for automotive logo identification is implemented in this work using MATLAB programming. The 68 vehicles logo was used to train the network. The application is tested using a noise image manually added to the symbols image.

After conducting an experimental investigation into the values of


Neural Network Training (nntraintool)

Neural Network

Input
Layer
W
b
Layer
W
b
Output

Algorithms

Training: Gradient Descent Backpropagation with Adaptive Learning Rate. (traingdx)
Performance: Mean Squared Error (mse)

Progress

```
Epoch:              0      312 iterations   1000
Time:                      0:00:06
Performance:        0.248  0.000994         0.00100
Gradient:           1.00   0.000186         1.00e-10
Validation Checks:  0      0                6
```

Plots

Performance (plotperform)
Training State (plottrainstate)
Regression (plotregression)

Plot Interval: 1 epochs

Opening Performance Plot

Fig. 6. Training process in MATLAB press "C" to continue running the program, and press "e" to exit the program. Your choice is …. …. …. …. ….

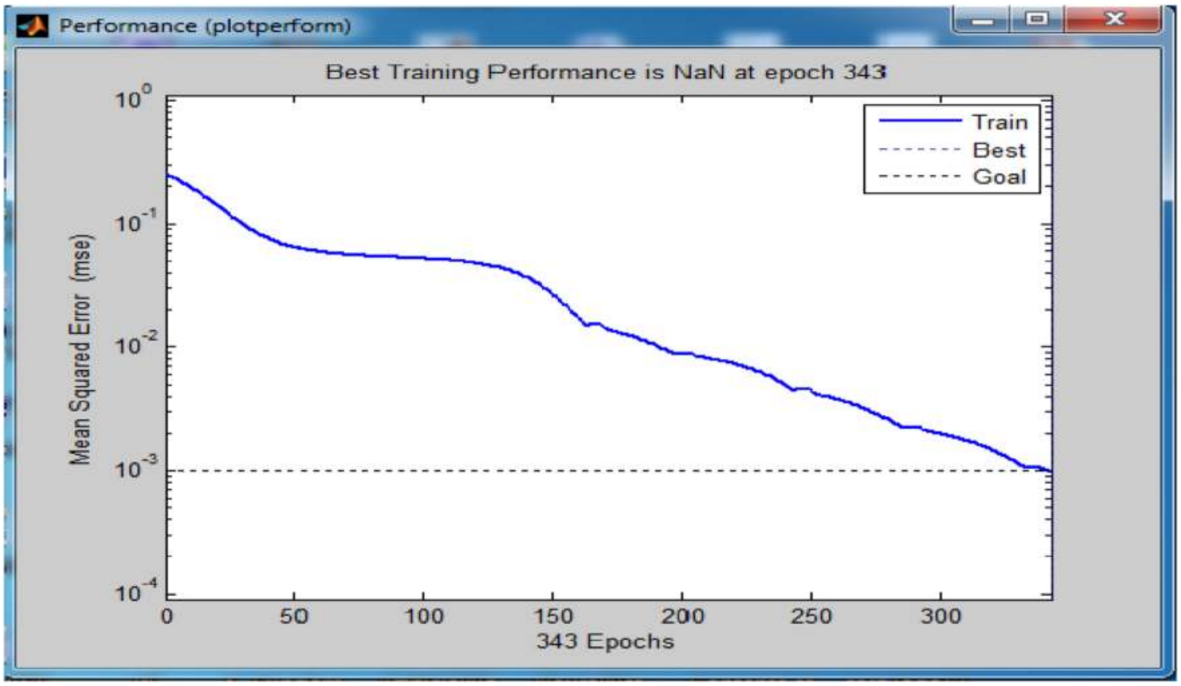


Fig. 7. The output curve of testing.

hidden neurons, we were able to achieve the following network convergence by selecting a suitable value of, escape from the local minima selecting, and the minimum value of Neuron number in the hidden layer to reduce process time and consider the stability of the convergence. The values of 50, 0.5, and 0.01 for the number of hidden layers and, respectively, were used in the implemented program.

The trial results show that the established program successfully tested the accuracy rate of logos (99.6%) and a failure in terms of incorrect logos (1.5% per cent). And the number of hidden layers, neurons, and (for) loops determine how fast the process takes. The program will run faster if you utilize the MATLAB built-in function Neural tools box, but you will have limited programming options when changing the parameters.

CRediT authorship contribution statement

Jabbar Majeed Sadeq: Paper written, simulation, out put figures, related work. **Brzo Aziz Qadir:** english editing, paper revised. **Hayder Hassan Abbas:** referencess arangement.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

[1] Y. Huang, R. Wu, Y. Sun, W. Wang, X. Ding, Vehicle logo recognition system based on convolutional neural networks with a pretraining strategy, IEEE Trans. Intell. Transport. Syst. 16 (4) (2015) 1951–1960, https://doi.org/10.1109/TITS.2014.2387069.

[2] L. Z, J.X. Ruikang Liu, Qing Han, Weidong Min, Vehicle Logo Recognition Based on Enhanced Matching for Small Objects, Constrained Region and SSFPD Network, Sensors, 2019.

[3] H.H. Abbas, A. Khashman, February, Evaluation of Geographical Information System and Intelligent Prediction of Solar Energy, vol. 1, 2016, pp. 106–112 [Online]. Available: http://www.iaras.org/iaras/journals/ijcsr.

[4] A. Khashman, H.H. Abbas, Acute lymphoblastic leukemia identification using blood smear images and a neural classifier, Lect. Notes Comput. Sci. 7903 (2013) 80–87, https://doi.org/10.1007/978-3-642-38682-4_10. LNCS, no. PART 2.

[5] Anderj Krenker,tanez Beter and Anderj Kos,faculty of Electrical Engineering, university of Ljubljana/slovinia.

[6] Introduction to the neural network by Eduardo Gasca Alvarez.egasca@ittoluca.edu.mx.

[7] D. Zheng, Y. Zhao, J. Wang, An efficient method of license plate location, Pattern Recogn. Lett. 26 (15) (2005) 2431–2438, Nov.

[8] S. smarasihge, Neural Network for Applied Science and Engineering, first. edition, Tylor and Francis, Network, 2007.

[9] R. Rojas, Neural Network, springer-verlage, Berlin, 1996.