Comparison of Two Transfer Functions in Feed-Forward Neural Network for Character Recognition phase in LPR System

Sawender Singh
Computer Sci. & Engg. Department
SVIET, Banur
Chandigarh India

Er. Manveer Kaur
Computer Sci. & Engg. Department
SVIET, Banur
Chandigarh, India

***Abstract* — Automatic license plate recognition system is an image processing technology used to identify vehicles by their license plates. Such systems require the localization of number plate area in order to identify the characters present on it.**

Feed-Forward Neural Network (FFNN) can be used to recognize the characters from images. This paper compares the two transfer functions in terms of their performance for the recognition phase: Radial basis transfer function and log sigmoidal transfer function, which are used to train the BP network. These two transfer functions are compared in terms of training performance, training time and number of training iterations to reach the optimal weights.

Keywords- Feed Forward; Radial Basis; Log Sigmoidal;

IJERT

I. INTRODUCTION

The process of vehicle License plate recognition requires a very high degree of accuracy when we are working on a very busy road or parking which may not be possible manually as a human being tends to get fatigued due to monotonous nature of the job and they cannot keep track of the vehicles when there are multiple vehicles are passing in a very short time. To overcome this problem, many efforts have been made by the researchers across the globe for last many years. A similar effort has been made in this work to develop an accurate automatic License plate recognition system. We have used MATLAB R2011b to obtain the desired results. We compared Feed forward network for two different transfer functions in character recognition phase. Results show that log sigmoidal transfer function works better than radial basis transfer function. The complete setup has been tested for 100 vehicles containing different License plates from different states. We get an overall efficiency of 93.75% for this system. Though this accuracy is not acceptable in general, but still the system can be used for vehicle identification.

Proposed Structure–The system presented is designed to recognize number plates from the front and rear of the vehicle. Input to the system is an image sequence acquired by a digital camera that consists of a number plate and its output is the recognition of characters on the number plate. The system consists of the standard four main modules in an LPR system, viz. License Plate Extraction, License Plate Preprocessing, License Plate Segmentation and License Plate Recognition.

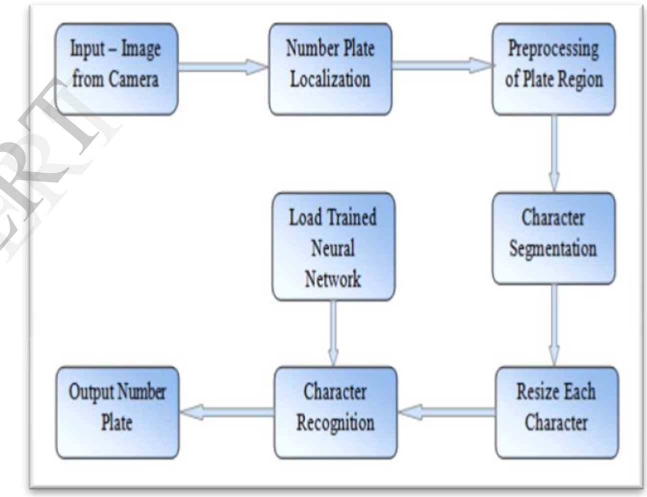
Input – Image from Camera
Number Plate Localization
Preprocessing of Plate Region
Load Trained Neural Network
Character Segmentation
Output Number Plate
Character Recognition
Resize Each Character

Figure 1: Structure Of LPR

The first task acquires the image and extracts the region that contains the License plate. The second task converts the color image to binary form and removes unwanted components. The third task isolates the characters, letters and numerals (total of 9 or 10 digits), as in the case of Indian Number Plates. The last task identifies or recognizes the segmented characters.

Our approach concerns stages of preprocessing, edge detection, filtering, and detection of the plate's position, and character segmentation and recognition. Single frame gray-level images are used as the only source of information.

II. CHARACTER RECOGNITION PHASE

A. *Feed-Forward Neural Network (FFNN)*

Neural network (NN) can be considered as non-linear statistical data modeling tool that can model almost any nonlinear relationship that may exist between inputs and

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

outputs or find patterns in data. These computational models are characterized by their architecture, learning algorithm, and activation function [1]. The feed-forward NN (FFNN) architecture is selected in this study. The FFNN consists of one or more nonlinear hidden layers. The hidden layers' activation functions are transfer functions that empower the network to learn the complex and nonlinear relationship between the inputs and the targets. In this architecture, a unidirectional weight connection exists between each two successive layers. A two-layer FFNN with transfer functions in the hidden layer and output layer can potentially approximate any function with finite number of discontinuities, provided a sufficient number of neurons exists in the hidden layer [2]. The Log-sigmoid transfer function and radial basis transfer function was used in hidden layer and the output layer.

B. *The Algorithm*

In MATLAB, a feed-forward backpropagation network is created using *newff* function. User needs to provide input argument such as input and output data, hidden layer and node size, node activation function, networks training algorithm and etc. The initial weight of the networks is randomly created by default, every time the *newff* is called. User also has a choice to initialize the random initial weight using *rands* command. The BP weight training can be directly executed using *train* function once the networks and the parameters are properly set. Here ms is column normalized array of each character and of size <1008 X 942>.

net =newff(minmax(ms), [50,942], {'logsig', logsig'},'trainscg');

III. TRANSFER FUNCTIONS

Training is the process of determining the optimal weights of the NN. This is done by defining a performance function (which is usually the mean square error between the network's output and the desired target) and then minimizing it with respect to weights. The minimization is performed by calculating the gradient using a technique called backpropagation which can be done in batch or incremental styles [1]. In this paper, we used the batch training style. The NN was trained using different transfer functions belonging to two classes described as follows:

A. Log-sigmoid transfer function (logsig):

The function logsig generates outputs between 0 and 1 as the neuron's net input goes from negative to positive infinity.

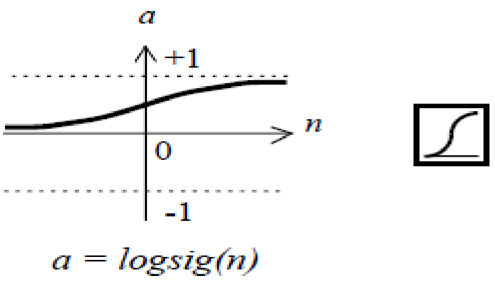

Figure 2: Log-Sigmoid Transfer Function

B. Radial Basis Transfer function (radbas)

The function logsig generates outputs between 0 and 1 as the neuron's net input goes from negative to positive infinity.

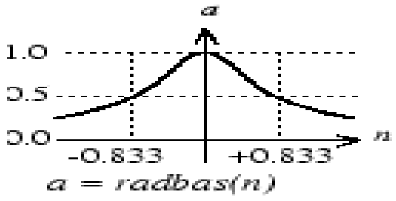

Figure3. Radial basis Transfer Function

IV. EXPERIMENTAL RESULTS

A. *Database*

The database used to train and test the systems described in this paper contains the binary images of characters as shown in figure 4. All the images were size normalized to fit in a 42x24 pixel box (while preserving the aspect ratio). Each character was normalized using column normalization in MATLAB. The normalized data was used as the inputs to the network. There were 942 target values corresponding to each training sample and size of the network was 1008 x 942.

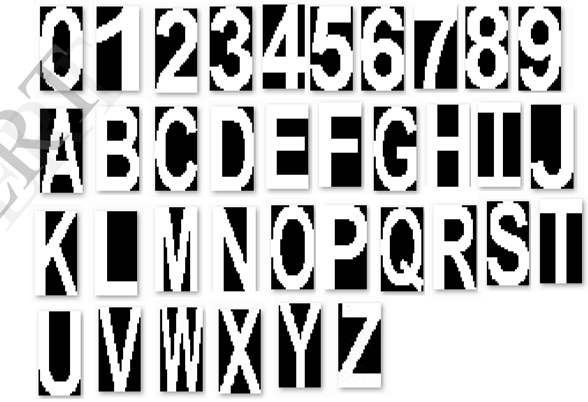

Figure 4. Binary characters

B. *Evalutation Criteria*

The BP estimation results were compared in terms of training performance. The training performance of the NNs was compared in terms of *1)* training time, *2)* number of training iterations to reach the optimal weights, 3) mean square error and 4) Regression rate.

C. *Parameter setup.*

The default value of MATLAB ANNs training parameters were used for all the algorithms except for the learning rate, goal and the maximum number of epochs.

net.trainParam.goal=0;
net.trainParam.epoches = 1000;
net.trainParam.lr = 0.01;

D. *Train and Test Strategy*

Same data set was used for both transfer functions. The network was trained with 100 % training dataset. Training was done using feed forward net with 1 hidden layer and at different number of hidden units. Input and output layers were having log sigmoid transfer function and radial basis

IJERT

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

transfer function alternatively. The mean square error (MSE) is the condition to terminate training of all the BP methods. MSE is originally set at 0.

We trained feed-forward networks with 50, 75,100,150, & 200 hidden units and observed the mean square error for both transfer functions. As we are using scaled conjugate training algorithm, over fitting will not happen as it does not updates its weight in negative gradient direction.

After the Training, network is simulated with different car images to calculate the accuracy.

E. Results

The performance of two different transfer functions is shown in Table1, and Table2. Table 1 shows the performance of logsig transfer function. Table 2 shows the performance of radial basis transfer function. These values were averaged over the 10 runs of the algorithms. The results were obtained on intel i5, 2.67 GHz processor with 3.0 GB of RAM.

Table1: Feedforward net Results with logsig transfer function

number of hidden units	Best Performance (mse)	Regression	Training Time (Seconds)	Recognition Accuracy
200	2.4816e-005	0.98824	802	93
150	2.4818e-005	0.98823	736	92
100	2.4832e-005	0.98823	618	91
75	2.5210e-005	0.98813	531	89
50	1.0076e-004	0.95768	457	72

Table2: Feedforward net Results with radbas transfer function

number of hidden units	Best Performance (mse)	Regression	Training Time (Seconds)	Recognition Accuracy
200	9.9954e-005	0.95171	702	87
150	1.8129e-004	0.91053	610	84
100	4.8152-004	0.73896	889	72
75	5.5159e-004	0.69284	754	67
50	7.5348e-004	0.53838	561	45

IJERT

With logsig transfer function, a lesser mean square error is achieved with less number of neurons. Comparing both transfer functions at 50 hidden units, we get less mean square error and more accuracy for logsig transfer function. With less number of hidden neurons, radbas performance was very low and accuracy of recognizing the character in car image was also very low. But as the number of neurons increases in radbas transfer function, we get improved mean square error and accuracy. Figure 3 and 4 shows the variation of mse with hidden units for both the transfer functions.

Among the logsig performances with different hidden units, least performance is obtained with less neurons and performance got improved with increase in hidden units. We got almost same mse for logsig transfer function after 100 neurons in hidden layer. With 200 neurons in hidden layer, we get overall mse for logsig transfer function was 2.4816e-005 and for radial basis transfer function, it was 9.9954e-005. With these performances, we compared both the networks which were trained using logsig and radbas transfer function, we checked 100 car images in recognition phase. Figure 5 shows the accuracy comparison of both the transfer functions with different hidden units.

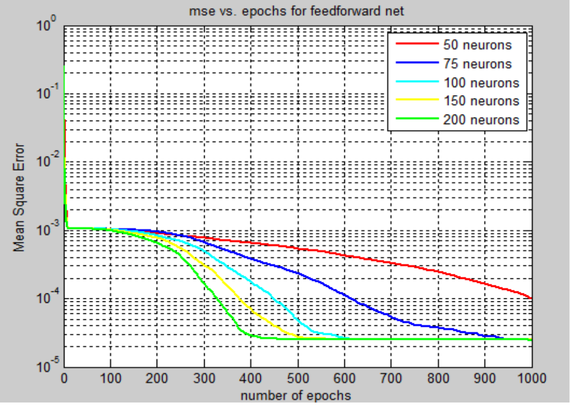

Figure 5: variation of mse with different neurons for logsig transfer function

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

With less number of neurons, accuracy of detecting the characters from the car image was very low for radbas transfer function, but for logsig transfer function, it was comparatively high. With increase in number of hidden units, accuracy got improved for radbas, but for logsig, it was almost constant after 100 hidden neurons. Overall, we can say that logsig transfer function performed well even with less number of neurons.

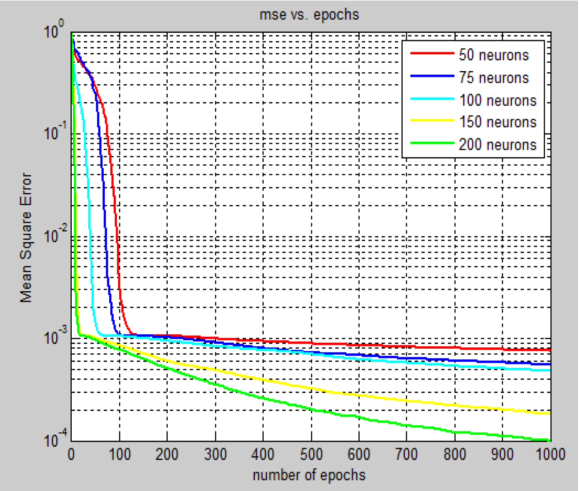

Figure 6: variation of mse with different neurons for radbas transfer function

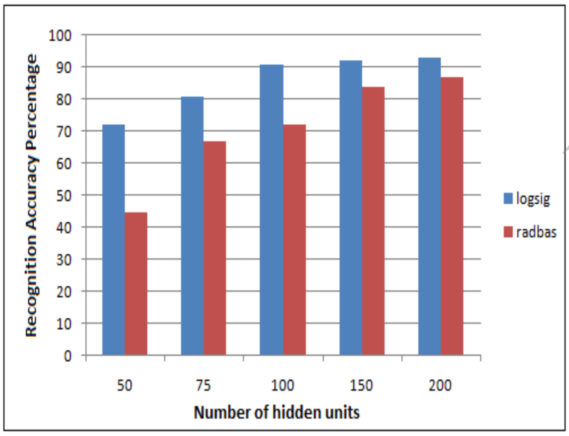

Figure 7: Comparison of recognition accuracy for logsig and radbas transfer function

V. CONCLUSION AND FUTURE WORK

We compared Feed forward network for two different transfer functions in character recognition phase. Results show that log sigmoidal transfer function works better than radial basis transfer function. The complete setup has been tested for 100 vehicles containing different License plates from different states. We get an overall efficiency of 93.75% for this system. There are a lot other transfer functions and many training functions like gradient descent, conjugated gradient etc. Their comparison will be studied next

REFERENCES

[1] J.-S.R. Jang, C.-T. Sun, and E. Mizutani, Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence, NJ: Prentice-Hall, 1997.

[2] G. Cybenko, “Approximation by superpositions of a sigmoidal function,” Math.Control Signals Systems, vol. 2, pp. 303-314, 1989.

[3] Xiaofeng Zhang, Fengchang Xu, and Yan Su, "Research on the Licnese Plate Recognition based on MATLAB," Procedia Engineering, vol. 15, pp. 1330-1334, 2011.

[4] Nicolas Thome, Antoine Vacavant, Lionel Robinault, and Serge Miguet, "A cognitive and video-based approach for multinational License Plate Recognition ," Machine Vision and Applications, Springer-Verlag, pp. 389-407, 2011.

[5] J. K. Chang, Ryoo Seungteak, and Heuiseok Lim, "Real-time vehicle tracking mechanism with license plate recognition from reoad images," The journal of super computing , pp. 1-12, 2011.

[6] A Kang, D. J;, "Dynamic programming -based method for extraction of license numbers of speeding vehicles on the highway ," International Journal of Automotive Technology, pp. 205-210, 2009.

[7] S.A. Daramola, E. Adetiba, A.U. Adoghe, J.A. Badejo, I.A. Samuel, T. Fagorusi “Automatic Vehicle Identification System Using License Plate”, Internatinal Journal of Engineering Science and Technology, vol.3, No.2, Feb 2011.

IJERT

(This work is licensed under a Creative Commons Attribution 4.0 International License.)