IJCAT International Journal of Computing and Technology, Volume 1, Issue 7, August 2014
ISSN : 2348 - 6090
www.IJCAT.org

An Approach to Recognize Characters using Neural Network in LPR System

[1] Sawender Singh, [2] Manveer Kaur

[1] Computer Science & Engineering Department, SVIET,
Banur, Punjab, 140601, India

[2] Computer Science & Engineering Department, SVIET,
Banur, Punjab, 140601, India

Abstract - Automatic license plate recognition system is an image processing technology used to identify vehicles by their license plates. Such systems require the recognition of characters from the plate image. Artificial neural networks are commonly used to perform character recognition due to their high noise tolerance. Feed-Forward Neural Network (FFNN) can be used to recognize the characters from images. The document is expected to serve as a resource for learners in pattern recognition, neural networking and related disciplines.

Keywords - **Feed Forward, OCR.**

1. Introduction

Optical Character Recognition, or OCR, is the process of translating images of handwritten, typewritten, or printed text into a format understood by machines for the purpose of editing, indexing/searching, and a reduction in storage size. Optical Character Recognition that would use an Artificial Neural Network as the backend to solve the classification problem.OCR is a field of research in pattern recognition, artificial intelligence and machine vision.

Problem of OCR is fairly simple:

Input: Individual character image after segmentation
Output: Computer readable version of input image.

This system is the base for many different types of applications in various fields, many of which we use in our daily lives. Cost effective and less time consuming, businesses, post offices, banks, security systems, and even the field of robotics employ this system as the base of their operations.

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. 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.

In this paper, we have presented a simplified approach to recognize the character using neural networks.

Proposed Structure of LPR– 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.

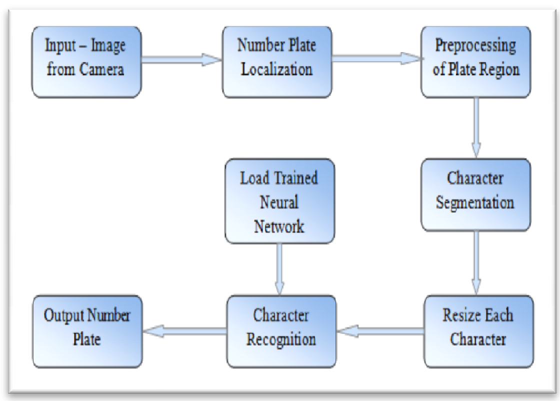

Figure 1: Structure Of LPR

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.

Output of third task is shown in the below image:

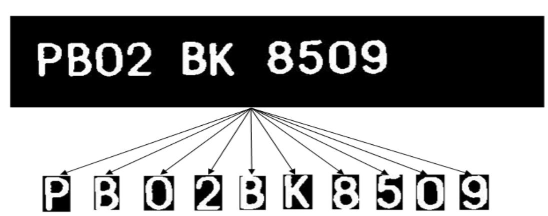

Figure 2: character extraction

IJCAT International Journal of Computing and Technology, Volume 1, Issue 7, August 2014
ISSN : 2348 - 6090
www.IJCAT.org

The last task identifies or recognizes the segmented characters. There are many different techniques available to recognize the character, but we chose neural networks because:

1. Neural nets learn to recognize the patterns which exist in the data set.
2. The system is developed through learning rather than programming.
3. Neural networks are flexible in a changing environment.
4. Neural networks can build informative models where more conventional approaches fail.
5. Performance of neural networks is at least as good as classical statistical modeling, and better on most problems.
6. Neural networks now operate well with modest computer hardware.

2. Character Recognition Phase

2.1 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 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. 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 was used in hidden layer and the output layer.

2.2 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');

2.3 Learning Process

The learning process plays important role for neural network. A neural network can learn the desired response from a set of samples and computes its output pattern, and if there is an error, weights or other parameters are to be adjusted to reduce the errors. Figure 3 shows that how unsupervised training occurs for neural network:

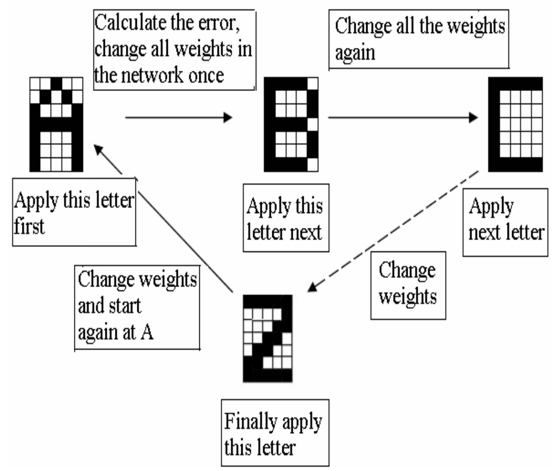

Figure 3: Learning process

3. Experimental Results

3.1 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.

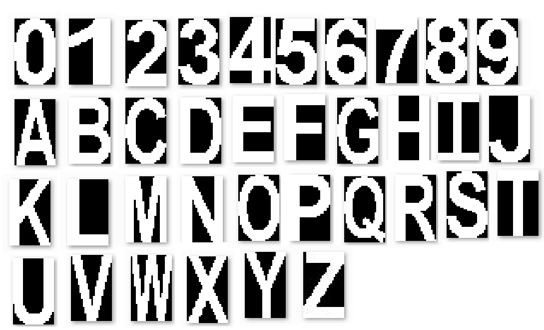

Figure 4. Binary characters

There were a total of 942 binary images out of which 421 images belong to alphabets and 521 images belong to numerals.

3.2 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;

3.3 How Recognition Will Work

So far we were able to segment each character from the binary image. Now each segmented character will be converted in to vector form and an already trained neural network will generate the confidence matrix which can be used to extract out the position of the letter in the database. Later the result can be displayed on a text file or on a message box. Figure shows our approach of character recognition.

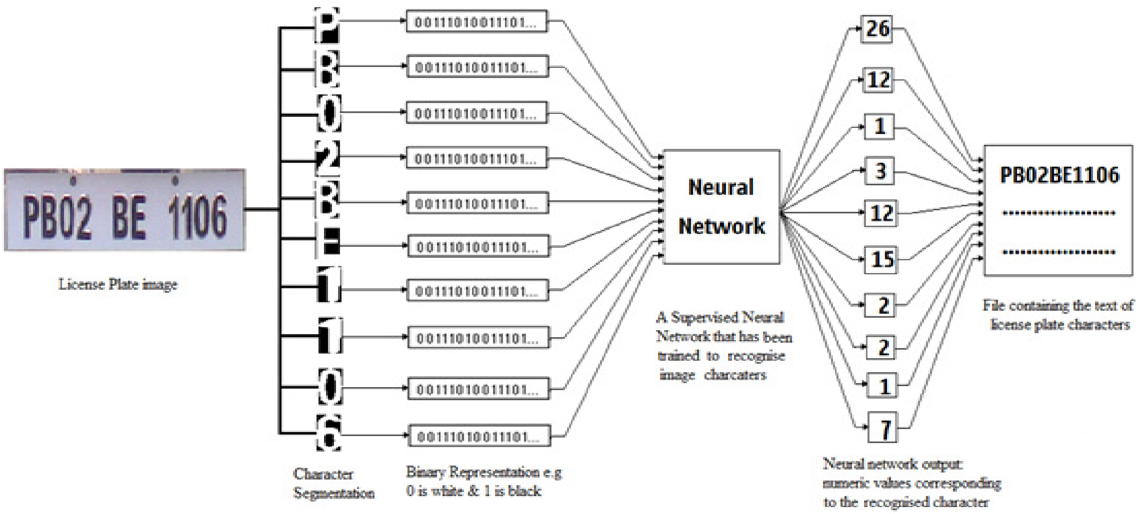

Fig 5. approach of character recognition.

D1. Candidate Score (ψ): This statistic is a product of corresponding elements of the weight matrix Wk of the kth learnt pattern and an input pattern I as its candidate. It is formulated using as follows:

$$\psi(k) = \sum_{i=1}^{x} \sum_{j=1}^{y} W_k(i, j) * I(i, j)$$

D2. Ideal Weight-Model Score (μ): This statistic simply gives the sum total of all the positive elements of the weight matrix of a learnt pattern.

$$\mu(k) = \mu(k) + W_k(i, j)$$

D3. Recognition Quotient: This statistic gives a measure of how well the recognition system identifies an input pattern as a matching candidate for one of its many learnt patterns. Its value lies between 0 to 1 and simply given by:

$$Q(k) = \frac{\psi(k)}{\mu(k)}$$

The greater the value of Q, the more confidence does the system bestow on the input pattern as being similar to a pattern already known to it.

3.4 Train and Test Strategy

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

Once the network weights and biases are initialized, the network is ready for training. The multilayer feed-forward network can be trained for function approximation (nonlinear regression) or pattern recognition. The training process requires a set of examples of proper network behavior—network inputs and target outputs.

The default performance function for feedforward networks is mean square error (mse)—the average squared error between the network outputs a and the target outputs t. It is defined as follows

$$F = mse = \frac{1}{N}\sum_{i=1}^{N}(e_i)^2 = \frac{1}{N}\sum_{i=1}^{N}(t_i - a_i)^2$$

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

With logsig transfer function, a lesser mean square error is achieved with less number of neurons. After the Training, network is simulated with different car plate images to calculate the accuracy.

3.5 Results

Table 1 shows the performance of feed-forward network. 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.

Table 1: Feedforward net Results

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

There is not a certain formula for calculating the number of hidden units required for weight convergence within the specified parameters. So hit and trail method used to find the optimum number of hidden units. We trained feedforward networks with 50, 75,100,150, & 200 hidden units and observed the mean square error for both transfer functions. The worst performance was 1.0076e-04 for 50 neurons and best performance was 2.4816e-005 for 200 neurons
.

4. Conclusion and Future Work

A simplistic approach for recognition of characters using artificial neural networks has been described. The advantages of neural computing over classical methods have been outlined. Despite the computational complexity involved, artificial neural networks offer several advantages in pattern recognition and classification in the sense of emulating adaptive human intelligence to a small extent. In this paper, we analyzed the character recognition phase with logsig transfer function. This work can be further extended to compare the results of different transfer functions like radbas, logsig, tansig etc.

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.