License plate recognition using Multi-Layer Perceptron Neural Network and Back-Propagation Algorithm

Shayan Asadpour*

*BSc Computer Software Engineer, shayan.asadpour@gmail.com

Abstract: *In this paper, we discuss a robust, fast, and low-error method for car plate recognition.*

License plate detection is one of the best ways to discover the identity of car drivers. This article discusses the car plate recognition (assuming that the plate on a car is previously located) and the way by which we separate different digits and letters. In the first section we will perform the initial processing, and in cases of possible existence of noise, we will improve the picture quality using filters like "The Gaussian Filter". In the second section we will separate the letters and digits; in order to achieve 8 single images. The aim of this section is to extract those portions of the image which contain letters or digits. The diagnosis is done using a "Multi-Layer Perceptron Neural Network" and the "Back-propagation" algorithm; which enjoys many advantages which we are to discuss shortly after. This Network is as much capable and robust that after iterating the test on 400 samples of car plates with different angles, shades, and dirt, the devised system will accurately recognize the letters by the rate of %94.25; and the time needed for completion of this is only 0.42 a second.

Keywords: License Plate Recognition, License Plate Detection, Gaussian Filter, Back-Propagation, Neural Network.

1. Introduction

Car plate recognition enjoys many advantages, among which are: finding the identity of offending drivers in highways and roads, measuring the speed of vehicles, controlling parking lots, and traffic. This technology has considerably contributed to traffic affairs.

Generally car plate recognition consists of 3 stages: 1- Locating the plate, 2- Segmenting the letters on a plate, and 3- Recognizing the letters. In this dissertation we are to discuss the 2nd and 3rd stages. We assume that the location of the plate has previously been detected using the either of the methods "Color Characteristic" or "Detecting the vertical edges" or "Morphology".

The stage of segmenting the letters can be done in a variety of ways: "Segmentation using histogram analysis method" and "Segmentation by detecting the joint characters method". In Car plates Issued in Iran (in Farsi letters and digits) using the histogram analysis method would return a high rate of error, because of the existence of screws and bolts, which do not appear to be located on a specific place on different plates. The method we used is "Segmentation by detecting the joint characters" which has a remarkably lower error in comparison with the other methods. In the plate recognition stage after we have segmented the letters, we employed the "Multi-layer Perceptron Neural Network" and "Back-propagation Algorithm". In the training stage the system needs only 3 numbers of times of training per letter, and its learning ability is %100; and when testing the taught samples the system would respond %100 correctly. In the generalization stage after testing the system with 400 samples %94.25 of the letters were accurately recognized.

2. Character Segmentation

The segmentation stage is one of the important sections before performing the recognition stage by the neural network. In fact, the segmentation stage is responsible for preparing good quality and noiseless images without any other distorting element, to be

handed over to the neural network for being distinguished and recognized. The aim of this section is to extract those parts of the image which contain letters or digits from those which do not contain any.

To achieve our goal we need to have a moderate threshold for all the images; thus we first balance the image threshold in the grayscale mode. The threshold of the image is calculated as in below (1):

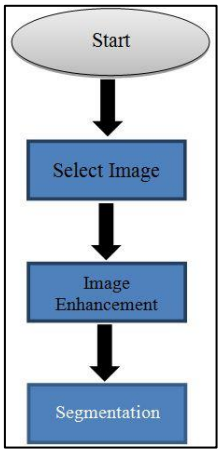

Fig. 1: Summary of Segmentation stages

$$g(x,y) = \begin{cases} 1 & if \quad f(x,y) > T \\ 0 & if \quad f(x,y) \leq T \end{cases} \qquad (1)$$

There are different stages for obtaining a suitable threshold:

1- We assign a value as the initial value for T.
2- To obtain an appropriate threshold we then divide the picture into two areas and name them R1 and R2.
3- R1 includes those pixels of the image which have a value greater than T, and R2 includes those which have a value less than or equal T.
4- Then the average of R1 is stored in the variable μ1 and the average of R2 is stored in the variable μ2.
5- The new threshold is calculated by formula (2):

$$T = \frac{(\mu 1 + \mu 2)}{2} \qquad (2)$$

6- We iterate the steps 2 to 5 to obtain a threshold less than the threshold introduced in the first step.

In the letter segmentation stage we used "Segmentation by detecting the joint characters" method. In this method firstly we transform the given image to binary. Then we find all the "1"s which are next to each other, group them, and then store the groups into a new matrix. This matrix includes all the shapes which are in a way joint to each other in the plate. In simple words, these are all the "1"s beside each other, that formed a white section of the image. One of the disadvantages of this method reveals when the plate is too dirty or too much lighted. In these cases, the detected characters are too many in number and in some instances up to 100 segmented components are extracted from the image. After experimenting and considering different aspects to troubleshoot this fault, we realized that in all cases all the digits have 40 to 120 rows, and 15 to 130 columns. Thus in such cases that too many elements are segmented from the image, we set a limitation for the program to find only the 8-needed elements out of the image.

An improvement method for the segmentation operation is eliminating the likely noises in the image. In some cases in order to prevent fault it is better to remove the boarder of the image if it exists. For instance if there is shade, or the car bumper is visible in the image it is better to improve and adjust the image. As mentioned above screws and bolts exist on plates (which do not appear to be located on a specific place); hence in order to enhance the segmentation and ameliorate the digit detection in the next stage, these bolts have been removed from the picture in a particular way.

The applied method for removing the bolt from the plate is based on color detection. In this method after reading the plate picture and before transforming it from RGB to binary we perform a color check. After experimenting on different plates with different conditions, the resulting solution was this: provided that a section in the plate is found with specific color degrees (red 140-190, green 160-190, blue 160-190) all the rows and columns of its pixels are stored in the matrix, and after transforming the image into binary, we substitute those sections with the color black. This way the bolt is removed from the image and becomes the color of background.

Fig. 2: A car plate image (before segmentation)

Fig. 3: The segmented car plate

The segmentation process has a remarkably low error, and after performing the test on 400 samples only %1.56 error was observed.

3. License Plate Recognition

3.1 Back-propagation Architecture

In the letter recognition stage we used "Multi-layer Perceptron Neural Network" and the "Back-propagation Algorithm". In this algorithm, the errors are fed-back to the previous layer by each of the iterations, to measure the error rate of the previous layer based on the next layer. For training the system for each image 3 times of training is sufficient. The segmented images are all cropped to the same size of 20 by 20. In addition, as in the training section 50 times of iteration is only needed for the response to reach to %100 of accuracy. The neural network here has a double-layer structure, consisting of one output layer and a single hidden layer.

The number of network inputs is the same as the size of the images. Because the input images are all 20 by 20 in size we have 400 neurons which are assigned as X1, X2, … , X10. Usually the number of neurons in the hidden layer is obtained experimentally. The number of neurons in the hidden layer is calculated this way (3) and they are named W1, W2, … , W290:

$$number\ of\ outputs + number\ of\ inputs \times \frac{2}{3} = number\ of\ neurons \quad (3)$$

$$25 + 400 \times \frac{2}{3} \approx 291 \quad (4)$$

Doing this calculation it was decided to use 290 neurons in the hidden layer.

In the output layer 25 neurons are used and named U1, U2, … , U25. The reason we chose 25 neurons in this section is that in Iranian car plates there exist 25 different letters and digits only. In addition, we used V1, V2, … , V290 as the bias for the hidden layer, and G1, G2, … , G25 as the bias for the output layer.

The Back-propagation neural network is a "Feed-Forward" network. Generally in these networks the signal is given to the inputs of the network; and the inputs would then send the signal to the hidden layer neurons. Then each neuron in the hidden layer calculates a value depending on its activity function, and sends it to the neurons in the output layer. At the end the output function calculates the values depending on its activity function and the final result is obtained in each of the output neurons. After each stage of training is terminated the output answers are compared to the goal values to determine the error in the output layer; and then on basis of the output layer error, the error for the previous layer (the hidden layer) is calculated.

After several experiments the "Tansig Activity Function" was chosen the middle layer activity function, and the "Linear Activity Function" was chosen the output layer function. In the "correct output selection" stage a competitive method was opted; i.e. after the calculations are done and the digits of the output layer are obtained, we took the maximum, and the first greatest number was chosen as the correct answer

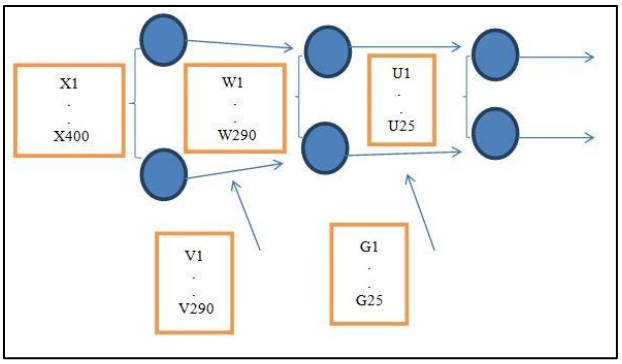

Fig. 4: The Neural Network Architecture

Also to update the values of scales (weights) in the hidden layer and output layer a value as "Momentum" is employed. At first variation of the scales is stored in the variables "outWeightChange" and "WweightChange". Next these values are multiplied by "Momentum" at the time of updating. After many experiments the value for "Momentum" was assigned 0.8. The "Learning Rate" (short LR) is as well after several experiments assigned 0.075.

3.2 The Employed Algorithm

1- Computation of the hidden layer's linear output:

$$I_h = \sum_{i=1}^{400} V(i) + x(i) * W(ij) \quad (5)$$

2- Computation of the hidden layer's activity function:

$$H_i = F(I_h) \quad (6)$$

3- Computation of the output layer's linear output:

$$Io = \sum_{i=1}^{25} G(i) + Hi * U(ij) \quad (7)$$

4- Computation of the output layer's activity function:

$$O = F(I_O) \quad (8)$$

5- Calculating the error rate in the output layer:

$$E_O = (T - Y) \quad (9)$$

6- Calculating the error rate in the hidden layer:

$$E_h = U_{ij} \times E_O \quad (10)$$

7- Updating the biases:

$$G_i = G_i + LR \times E_O \quad (11)$$

$$V_i = V_i + LR \times E_h \quad (12)$$

8- Updating the scales (weights) using the momentum:

$$WOldWeight = U_{ij}$$

$$U_{ij} = U_{ij-1} + \mathrm{LR} \times E_O \times H_i$$

$$WweightChange = U_i - Woldweight$$

$$U_{ij} = U_{ij} + (WweightChange \times Momentum)$$

(13)

$$OutOldWeight = W_i$$

$$W_{ij} = W_{ij} + LR \times E_h \times X_i$$

$$outWeightChange = W_i - OutOldWeight$$

$$W_{ij} = W_{ij} + (outWeightChange \times momentum)$$

(14)

Another point in a Multi-layer Perceptron Neural Network is choosing the initial scales (weights) and biases that are used. Initially all the scales (weights) for the output layer are chosen zero, and for the middle layer the scales are chosen from small random numbers between -0.05 to 0.05. All the biases are chosen the same at first.

Not only has the recognition of the segmented letters in the training stage returned % 100 correct responses, but also the test stage has returned %100 accurate results. In the generalization stage provided that the image of the plate is flawless, clean, and without shade the result is still %100 correct.

One of the characteristics of this system is being angle-independent. In other words, if the image is of good quality and the plate is shade-less, no matter what angle the plate has in the image, the result is again %100 exact. In some cases where the image was defected we observed a tiny percent of error. Generally, in the recognition stage the system has %94.25 of accuracy on a basis of 400 sample tests.

4. Some of the results

Fig. 5: A shady car plate

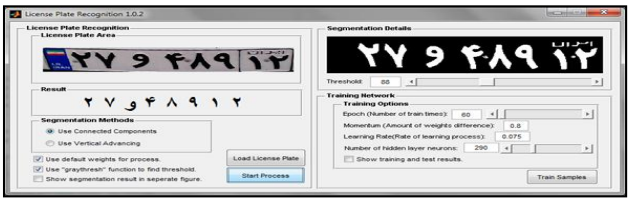

Fig. 6: An angular car plate

Fig. 7: A perfect car plate

5. Conclusion

The Car plate recognition systems are one of the most useful applications of artificial intelligence. Hence in this project it was best tried to have the least error possible. As mentioned before we used the Multi-Layer Perceptron Neural Network and the Back-propagation Algorithm; and the results are completely successful and the devised system is capable of recognizing the given data by a highly remarkable accuracy. In the segmentation section which is not dependent on the angle of the plate only a %1.56 of error was observed among 400 sample tests.

The best activity functions like "Tansig" were employed for the hidden layer and the "Linear Activity" function was also applied for the output layer. The momentum was chosen 0.8 and the learning rate was obtained 0.075. Also for the hidden layers weights were chosen randomly between -0.05 and 0.05. The results of this project have been extremely satisfactory and to state again in 400 tests %94.5 of them were correctly recognized. I hope to be able to add a Plate Locating System in the future to the present project; and expect to introduce the system to the market.

Acknowledgments

I would like to express my highest gratitude to Mrs. Abnavi, my neural network course professor for her valuable assistance and advice, and responsible supervision of this dissertation.

References

Papers from Conference Proceedings (Published):

[1] Devinder Singh and Baljit Singh Khehra, "DIGIT RECOGNITION SYSTEM USING BACK PROPAGATION NEURAL NETWORK", International Journal of Computer Science and Communication Vol. 2, No. 1, January-June 2011, pp. 197-205.

[2] W³odzis³aw Duch and Norbert Jankowski, "Transfer functions: hidden possibilities for better neural networks. ",ESANN'2001 proceedings - European Symposium on Artificial Neural Networks Bruges (Belgium), 25-27 April 2001, D-Facto public., ISBN 2-930307-01-3, pp. 81-94.

[3] Jamal M. Nazzal, Ibrahim M. El-Emary, and Salam A. Najim"Multilayer Perceptron Neural Network (MLPs)
For Analyzing the Properties of Jordan Oil Shale", World Applied Sciences Journal 5 (5): 546-552, 2008 ISSN 1818-4952© IDOSI Publications, 2008.

[4] Li-Shien Chen, Yun-Chung Chung, and Sei-Wan Chen Shyang-Lih Chang,"Automatic License Plate Recognition,"IEEE *Transactions on* IntelligentTransportation Systems, vol. 5, no. 1, pp.42-53, March 2004.

Papers Presented at Conferences (Unpublished):

[5] Yungang Zhang and Changshui Zhang, "A New Algorithm for Character Segmentation of License Plate", June 2003.

[6] CheokMan and Kengchung, "A High Accurate Macau License Plate Recognition System", 2008.

Books:

[7] R.C. Gonzalez., R.E. Woods, and S.L. Eddins, *Digital Image Processing Using MATLAB,* Publishing House of Electronics Industry, pp. 280-281,2005.