Text Recognition using Image Segmentation and Neural Network

Ammar A. Radhi

ABSTRACT
The fast development in worldwide technology makes it essential for us to find solutions for some challenging problems; one of these is the recognition of the important details from images. Text recognition is a process that converts the text that any image contains to something that is recognizable for our modern devices e.g., our smart phones. In this paper an efficient algorithm for the detection of text from an image is proposed. This methodology includes the application of a powerful segmentation method that labels the letters in the image very efficiently, and then the mean of the coordinates of each letter within the image is used after the application of the segmentation method in order to sort the letters correctly, moreover, the algorithm includes the introduction of a new line detection method based on the mean of the labels, and finally some modifications on the work of the autoencoder neural network is proposed in this paper. The transformation of the text in the image to text document is achieved with high accurate result; the results show a complete recognition of the text in wide varieties of images. Examples of the application of this algorithm are for the detection of the text in image with many lines, road sign images and car plate number recognition are shown in this paper.

General Terms
Text recognition, Neural network, Image segmentation

Keywords
Neural network, Image segmentation, Car plate number detection, Road sign detection.

1. INTRODUCTION
Text recognition from an image is consider a very important task because of the wide variety application of modern devices which needs the identification of an image's details, for example, the recognition of someone's signature, transaction in some banks are now undergoes with filling up a paper and then the rest of the work is upon the machine to recognize the details, automatic correction of multi choices questions in some internationals exams, the conversion from pdf file format to word document, car plate number detection and, the identification of the address from the road signs. Early methods of the recognition were mainly based on template matching and structural characteristics [1]. This template is either formed artificially, or selected from samples. Since the number of samples used in the template is normally limited and cannot absorb all the potential cases, thus the ability of these methods is limited. Then the recognition is moved to more practical solution by the introduction of the artificial neural network, the use of this network allow the training of the recognition system by a lot of possibilities of the same letter or number, this will cover more cases, such as the shape of the letter in handwriting by different hand [2][3]. In this work the image will be first subjected to some feature extraction process, this process involved some pre processing and post processing procedure in order to detect the position of the text in the given image, and the extracted text is then fed to a neural network, which is trained with different cases of each alphabet letters and numbers, such cases include rotation, distortion, noise and other possibilities. The proposed method includes the introduction of an efficient segmentation method for detecting and labeling letters, and then a new line detection method is applied. A modified neural network with both supervised and unsupervised training is used for recognizing the letter extracted from the image, while the final decision is after a correlation operation of the output of the neural network with a reference one in a special template. The method shows a very good result in the recognition of the text from images with many lines, images with roads' name and car plate number recognition.

2. LITERATURE RIVIEW
Several algorithms were introduced to detect the characters from the image, in [4]; the authors introduced an optical character recognition in order to convert any scanned document to a format that is recognizable for the computers. In [5], an algorithm for text spotting and recognizing text in natural science image is performed. However, many challenges are accompanying the recognition from such images i.e., the text in such images is widely variable considering its layout and fonts. The neural network used in this task was trained by complete words from dictionary instead of letters. Many methods followed two approaches in the detection, first text detection and then word detection [6], in text detection, the region of the word is spotted and then the characters of this word is detected. Mainly, the recognition methods are divided to statistical methods, artificial neural networks, kernel methods and multiple classifier combination methods. The statistical method is based on Bayes decision rule which is even subdivided into parametric ones and non parametric ones, the non parametric ones such as Pazern window and k-NN rule, in this method all the samples are stored, and therefore it is not suitable for real time applications [7] [8]. The recognition using neural network could be accomplished by adjusting the weight of the network branches iteratively and according to the minimum mean square error rule. The convolution neural network in [9] [10], has reported a high success rate in the recognition of the characters. The kernel methods normally map the data to a higher dimensional space, the data in this space exhibits linear pattern, and could be linearly separable in its new representation [11]. The kernel method is only applied to small set of problems that's because of its high complexity of the training and execution. This complexity could be reduced by the use of neural network first in order to select two nominated classes and then the kernel method could be applied [12].

Finally, the multiple classifier combination is introduced for increasing the accuracy maintained by a single classifier [13]. Horizontal combination of classifiers is mainly used for high accuracy purposes, whereas the cascaded combination is used

for speeding up the large set classification.

Overall, the method used in this paper combined the segmentation methods used normally in image processing with the use of neural network to optimize the recognition of the text; the algorithm is simple and efficient regarding the recognition issue with accurate results in comparison to the mentioned methods.

3. LETTER EXTRACTION

In order to spot the position of the text in the image some morphological processes are applied.

3.1 Preprocessing

First the image is converted from an RGB image to grayscale image; this means removing the colors of the image and converting the matrix of the image from three dimensions to two dimensions only. Then the grayscale image is converted to binary image. Figure (1) shows the image in RGB and the image in binary format.

(*a*) (*b*)

Fig 1: (a) Original image. (b) Original image in binary format.

the image in figure (1)(b) is then applied to some area opening process, which removes any unimportant details that occupies fewer than or equal to 100 pixels, this will remove some unnecessary details from the image.

3.2 Segmentation

Segmentation is an important process in image recognition; it's referred to the isolation of the position of the words within the image, and then the extraction of each letter individually. This could be accomplished by the application of the formula in equation (1) [14]:

$$X_K = (X_{K-1} \oplus B) \cap A \quad \text{..... Eq.1}$$

Where: k : is 1, 2, 3… until $X_K = X_{K-1}$.

X_K : is a matrix of the first non zero component in the given image since the image is in binary format.
B: is a 3*3 square structure matrix.
A: is the original image in binary format.

The iterative application of equation (1) will result in labeling the image's objects with different labels, in case of the image in figure (1) (b) each letter will be individually labeled with different number. Equation one is implemented using matlab with the following steps:

1. Detect a non-zero component in the binary image.
2. Create a new matrix (X) that contains only the non-zero component detected from step 1. The rest of the values in the matrix will be zeros.
3. Perform the following recursive function $X_K = (X_{K-1} \oplus B) \cap A$ for k=1,2,3,.. until $X_K = X_{K-1}$.
4. The result in step 3 represents the first label.
5. Recreate the original image matrix (A) to be (C) by making each number one that belongs to the first label zero in order to detect the next label.
6. Detect the next non-zero component from the image matrix created in 5.
7. Repeat the function $X_K = (X_{K-1} \oplus B) \cap C$, if $X_K = X_{K-1}$ then result will represent the second label.
8. Repeat step 6 until there's no other number one to detect.

The result of this algorithm is labeling the contents of the image. After the removal of the unimportant details from the image by the area opening operation, the remaining objects are the letters and numbers, thus only letters and numbers will be labeled by equation (1). Sometimes the labeling order of the letters is not arranged correctly, that's because the labeling operation depends on the first appearance of the object in the image and from left to right. The next section discusses how to overcome this problem.

3.3 Detection of the Beginning of Lines

The proposed algorithm of segmenting the image in equation (1) moves through the image and label the content based on their first appearance and from left to right of the image. If the lines are all started from the same x coordinate the label will be as shown in the figure (2).

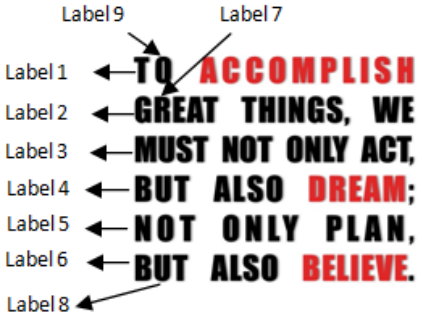

Fig 2: Labeling the content of the image

The correct form of labeling required that the next label after label 1 should be letter 'O' instead of letter 'G' of the next line. Otherwise the final output will be 'TGMBUT' instead of 'TOACCOMPLSH'. In order to fix this, the labeling of the algorithm remains the same and a threshold is calculated between each two line, first the mean of the pixels' y-coordinate of the first label (of letter 'T') is calculated, then the same mean is calculated for the next label (of letter 'G') and the average of the two mean is calculated using equation (2),

$$\text{Threshold 1} = \frac{(\text{mean of label 1} + \text{mean of label 2})}{2} \quad \text{.....Eq(2)}$$

The same process is repeated between label 2 and 3 until no other label is left. The separation of the lines is performed by calculating the mean of each letter, and then a comparison is made between the mean values of each letter to the thresholds calculated. Table 1. Shows the means of the letters of the labels from 1-9. And the threshold calculated.

Table 1. Shows the means of the letters and the threshold between each two lines

Mean of the Letters	
1	27.9685
2	74.3678
3	116.7003
4	160.4532
5	203.5872
6	246.8415
7	73.9048
8	246.5290
9	30.6651

Threshold	
1	51.1682
2	95.5341
3	138.5767
4	182.0202
5	225.2143
6	160.3731

The selection of the priority of extracting these labels from the image is based on a comparison with the threshold shown in Table (1). Each label has a mean of lower than 51.1682 will be selected first. It's clear that label 1 which represents the letter 'T' has a mean y- coordinate of 27.9685, which is lower than the first threshold, as well as the letter 'O' which is labeled as label 9 and has a mean of 30.6651, thus the algorithm selects the letter 'O' instead of letter 'G' which has a higher mean than the first threshold.

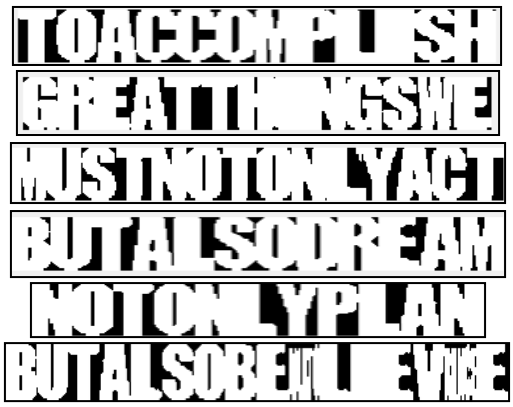

Fig 3: Detection of each line's beginning and sorting labels

Figure (3) represent the cells that carry the letters of the sentences in the tested image.

3.4 Resizing

Prior to feeding these letters to the neural network a resize operation of each letter is necessary, since the letter extracted from different images may vary in dimension, therefore there's a high probability that it will not match the size of the target set that the neural network has been trained with. The target set used here has an individually dimension of '42*24', thus each letter extracted is mapped to 42*24.figure (4) shows a sample letter before and after the resizing operation.

(*a*) (*b*)

Fig 4: (a) letter extracted from source image (b) letter resized

4. NEURAL NETWORK

Neural network is a tool used to capture the relationship between the input and the output, it is designed so that it will acquire knowledge and develop itself to be smarter in order to produce the appropriate output based on the input and the training set. Complex mathematical calculations are performed in this network and it saves the knowledge acquired in what is known as the weight. The weight belongs to the braches that connect the neurons between a specific layer and the next layer. The reason behind the use of the neural network in the detection of any digit or number is that a letter extracted from one image might be different from a letter extracted from another one, for example, it could be distorted, rotated, thinner, or wider. By taking the case of the detection of the letter R which is shown in figure (5) (a) and figure (5) (b),

(*a*) (*b*)

Fig 5: Road's sign with different capturing mode

The extracted letter R from the image in figure (5) (a) and (b) is shown in figure (6),

(*a*) (*b*)

Fig 6: (a) the letter extracted from the image in figure (6) (a), (b) the letter extracted from the image in figure (6) (b).

From figure (6) it's clear that the two extracted letters from the different images above have different appearance, thus it's necessary to have something that could decide that the two letters are in fact the same. This represents an easy task for the well trained human brain but it's trivial for the computer, that's why it's important to train something that should think the same way of the human brain. If the neural network is well designed, for instance, by training this network with different and wide range of possibilities of each letter, it can decide the correct letter efficiently. Figure (7) shows some of the training set that could be used to train the network relating the letter R.

Fig 7: Some of the training set of letter R

4.1 Neural Network and Data Compression

The neural network toolbox used here is called autoencoder. The autoencoder is a function for training deep neural network. This neural network forms multi layers, which compresses the input data from high dimension to a simpler form of a compressed version Figure (8) represents the architecture of this neural network with its layers [15].

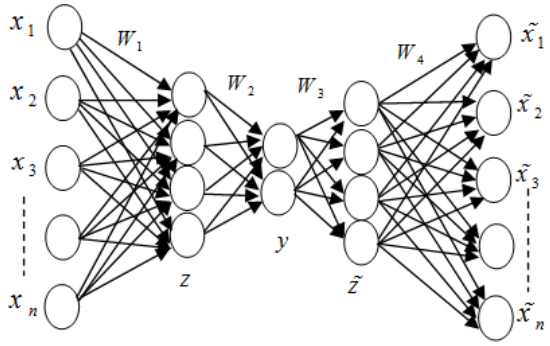

Fig 8: Architecture of the Neural Network

x_i - Input data

$W_{1,2,3,4}$ - Weight of the branches that connects layers

The encoder first converts the input to a hidden representation; this stage represents a feature extraction stage where the encoder extracts the features from the data, then forms a compressed version of this data, while the decoder is responsible for the reconstruction of the original data. Figure (9) shows the process between the encoder and the decoder.

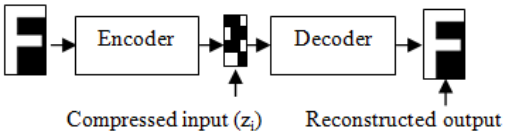

Fig 9: Example of how the autoencoder works.

4.1.1 Training the neural network

The training of the neural network is very important process; this operation involves with familiarizing the network with the type of the data used and enables the network to adjust itself in order to recreate the data again. If we supposed that we have a training set of input images $\{x_1, x_2, ..., x_n\}$ each image in this data set has a high dimension; the encoder maps these images to another form, suppose it's $\{z_1, z_2, ..., z_n\}$, Where the z_i's has lower dimension than the x_i's .The autoencoder neural network performs encoding of the input data twice, first the input data set x is mapped to the data set z using equation (3) [16]. ,

$$z_i = W_1 x_i + b_1 \quad \text{.....Eq(3)}$$

Then the compressed version of the input data set is encoded again using equation (4),

$$y_i = W_2 z_i + b_2 \quad \text{.....Eq(4)}$$

The decoder then maps the compressed data y back to its first compressed form z using equation (5),

$$\tilde{z}_i = W_3 y_i + b_3 \quad \text{.....Eq(5)}$$

This data set is decoded again to recreate the original data set using equation (6).

$$\tilde{x}_i = W_4 \tilde{z}_i + b_4 \quad \text{.....Eq(6)}$$

The mapping between the inputs, the hidden layers, and the output undergoes back and forth to adjust the weight of each branch in the network according to the training set. A Minimum Mean Square Error (MMSE) function is used by the autoencoder to adjust the branches' weight; this operation could be achieved by the iterative application of equation (7) to decode the data set from y to z [16]. .

$$J(W_1, b_1, W_2, b_2, W_3, b_3) = \sum_{i=1}^{n} (\tilde{z}_i - y_i)^2 \quad \text{..... Eq(7)}$$

$$= \sum_{i=1}^{n} (W_3 y_i + b_3 - y_i)^2$$

$$= \sum_{i=1}^{n} (W_3 (W_2 z_i + b_2) + b_3 - y_i)^2$$

$$= \sum_{i=1}^{n} (W_3 (W_2 (W_1 x_i + b_1) + b_2) + b_3 - y_i)^2$$

Equation (8) is used to reconstruct the data from z to x which is the original form of the data.

$$J(W_2, b_2, W_3, b_3, W_4, b_4) = \sum_{i=1}^{n} (\tilde{x}_i - \tilde{z}_i)^2 \quad \text{..... Eq(8)}$$

$$= \sum_{i=1}^{n} (W_4 \tilde{z}_i + b_4 - \tilde{z}_i)^2$$

$$= \sum_{i=1}^{n} (W_4 (W_3 y_i + b_3) + b_4 - \tilde{z}_i)^2$$

$$= \sum_{i=1}^{n} (W_4 (W_3 (W_2 z_i + b_2) + b_3) + b_4 - \tilde{z}_i)^2$$

The network will performs an iterative process that will minimize the difference between the input and the estimated output, in another word, the weight of the branches is adjusted iteratively to give a minimum difference between the input and the estimated output in each stage and according to equation(7) and (8).

4.2 Neural Network Toolbox

In this work a neural network with two hidden layers has been trained to classify the image using this toolbox. The two hidden layers are trained separately by the autoencoder, then a softmax layer is trained, and finally the layers are gathered to form a deep neural network.

4.2.1 The first autoencoder

The first hidden layer consists of 100 neurons in which the weight of this layer is adjusted by using equation (7). Figure (10) shows the first autoencoder [17].

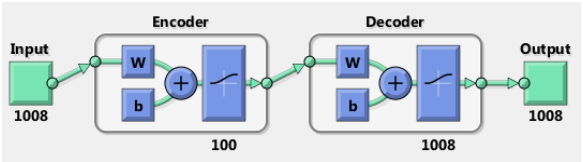

Fig 10: The first autoencoder

The features extracted in the first autoencoder represent the curls and stroke patterns from the letter image. The training images loaded to the encoder in the first autoencoder have a dimension of 42*24 for each one which forms 1008. The extracted feature after passing through the first encoder has a dimension of 100 for each image in the training set instead of 1008, and thus the learned features represent a compressed version of the input data.

4.2.2 The second autoencoder

The second autoencoder also has an encoder and decoder, the same method of training is used here, however, the only

difference here is that the training set used in this step is the features that were extracted from the first stage. In addition, instead of using 100 neurons in the hidden layer of the second autoencoder only 50 are used. As a result, the encoder in this stage will learn an even compressed representation of the input data. Figure (11) shows the second autoencoder [17].

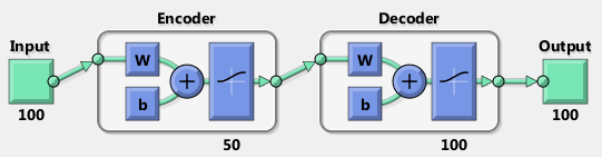

Fig 11: The second autoencoder

This data is used to train the last layer and then the network will be able to classify each digit and number.

4.2.3 The final softmax layer

This stage is necessary to classify the compressed data from the second encoder of the second autoencoder. The dimension of the data is now 50 as a result of the two encoding processes instead of 1008. Figure (12) represents the softmax layer [17].

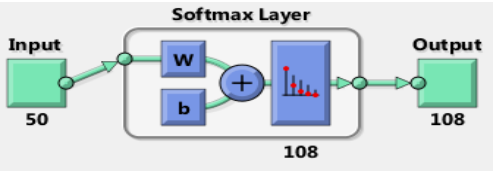

Fig 12: The softmax layer

4.2.4 Stacked autoencoder

The two encoders should now be placed together with the softmax layer to form the deep network [17].

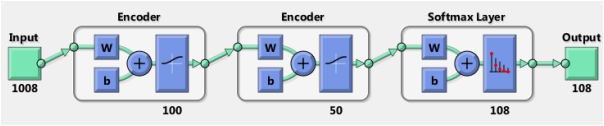

Fig 13: The autoencoder neural network

4.2.5 Forward propagation

The forward propagation is performed first, this involves an unsupervised learning for the neural network, which means that the two encoders and the softmax layer should be trained without a target as shown in figure (14), only the input is applied to the neural network and as a result the neural network will be trained to find a structure or a relationship between the inputs [17].

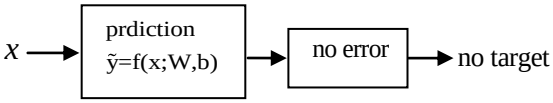

Fig 14: Unsupervised learning.

4.2.6 Back propagation

The final result of the neural network could be further improved by retraining it with back propagation. In back propagation the neural network is trained in a supervised way, this involves training the network for every input with corresponding target [17]. This will provide a decision upon every input. Figure (15),

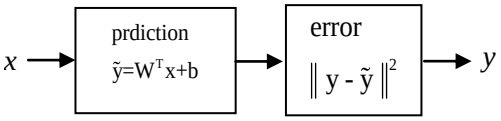

Fig 15: Supervised learning.

4.3 How this Neural Network Work

A modification has been made on the work of this neural network, normally this neural network will regenerate the original image at the output as shown in figure (9), however ,here it is modified to work differently. After the training of the network is completed, the input to the softmax layer will be the compressed version of letters, let's say that the input is the letters shown in figure (3), where each letter has a dimension of 50 individually.

The output of the softmax layer will be a matrix of (108*number of input letters), the row 108 is associated with the number of letters in alphabet and numbers from 0 to 9 which is 36, with each one, two extra models are added for the training purpose of the neural network, then 36*3=108.

The column (number of input letters) represents the letters that the neural network needs to recognize. If the neural network detects a matching between the input and the training set, it will give a number out of 1 to indicate the measure of this matching.

In case the input is the word 'TOACCOMPLISH' in figure (3) this word has 12 letters, if we apply it to the softmax layer then the output will be a matrix of 108*12, in the column of this matrix there will be only one element that is not zero and high enough to identify that there is matching between the letter associated with this column and with a letter exists in the training set.

After the identification of the letter by the neural network, and as a double check procedure, a correlation operation is applied between the letters extracted and the letters exists in a template, this template has the original standard images of the alphabet letters and numbers that are used to detect the letter. The final stage will be writing the letters recognized from the images to a text file.

Figure (16) shows the flowchart of the recognition process.

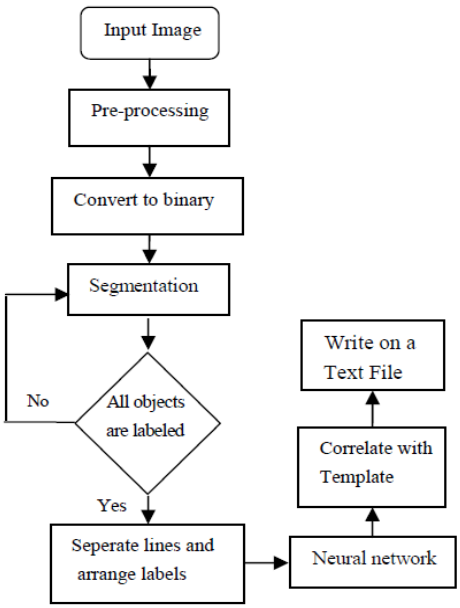

Fig 16: Flow chart of the algorithm.

5. THE RESULTS

The algorithm stated in this paper is tested with different examples, images with many lines, road sign images, and car plate numbers. The example in figure (17) shows the recognition of the text from the first image; this involves the detection of the beginning of each line and the recognition of each letter, then transferring the text to a text file, which will make it accessible and easy to understand by any machine. The number of iterations used in this recognition was 1000 epochs.

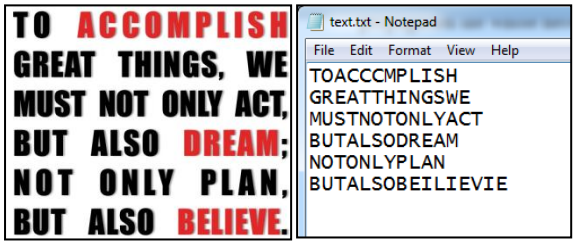

Fig 17: Example (1) recognition of the text from image containing many lines.

The result in figure (17) shows that the letters were preserved and retrieved correctly from the image, and the algorithm performed well relating to the separation of the lines from each other.

Moreover, the algorithm has been tested in the recognition of the text in the sign roads; figure (18) shows the second example.

Fig 18: Example (2) recognition of the first road sign.

Figure (19) (a) presents the result of the first part of the algorithm, the proposed segmentation method and before applying the data to the neural network. The letters extracted are all the same as in the image and the separation between the lines is preserved.

(a) **(b)**

Fig 19: (a) the extracted letters from the image (b) the output letters from the neural network.

Figure (19) (b), represents the output letters from the neural network which resample's the shape of the letters that exists in the template. Figure (21) to figure (30) show other examples of the application of this algorithm in recognizing the text in the road signs.

The number of iterations used in these examples was 400. The minimum mean square error converges from 1.9 to $5.5*10^{-2}$ within the first 100 epochs as shown in figure (20), which will give the best result.

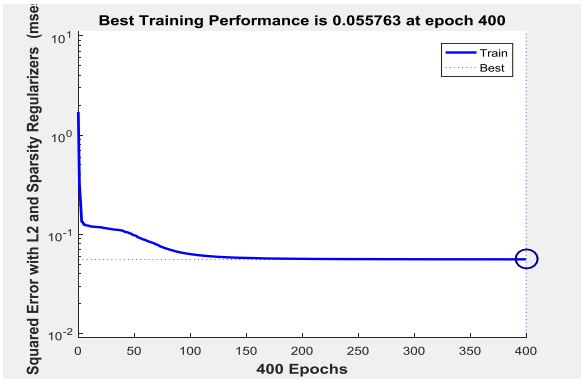

Fig 20: Best training performance curve

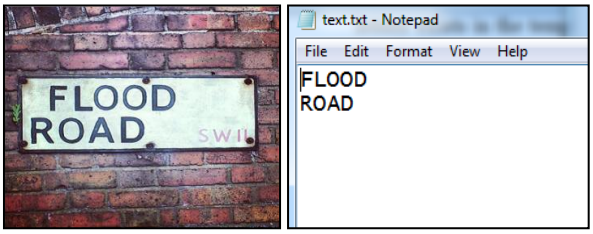

Fig 21: Example (3) recognition of the second road sign.

(a) **(b)**

Fig 22: (a) the extracted letters from the image (b) the output letters from the neural network.

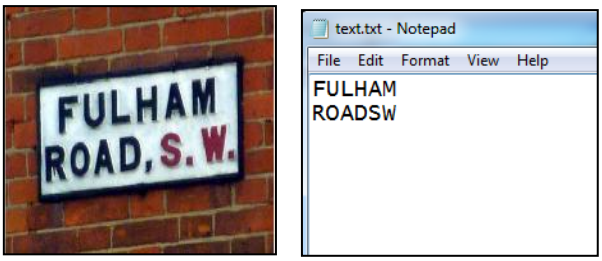

Fig 23: Example (4) recognition of the third road sign.

(a) **(b)**

Fig 24: (a) the extracted letters from the image (b) the output letters from the neural network

The algorithm in this paper has been tested also in the recognition of numbers from the car plates.

Fig 25: Example (5) recognition of the first car plate number

(a) (b)

Fig 26: (a) the extracted letters from the image (b) the output letters from the neural network.

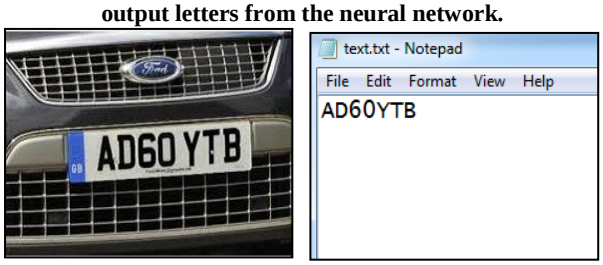

Fig 27: Example (6) recognition of the second car plate number

(a) (b)

Fig 28: (a) the extracted letters from the image (b) the output letters from the neural network.

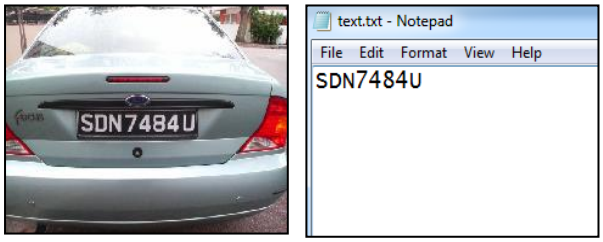

Fig 29: Example (7) recognition of the third car plate number

(a) (b)

Fig 30: (a) the extracted letters from the image (b) the output letters from the neural network

5.1 Comparison of the Results

Two comparisons have been made about the result, first whether the neural network was not a part of the algorithm, and the second one is what happens if the number of epoch in the program is increase or decrease

5.1.1 If the neural network was not a part of the recognition process.

In case that the neural network wasn't considered as a part of this algorithm, the result would be random and not that accurate unless the letters extracted from the image is very much similar to the reference letters used. This theory has been applied to the image in figure (17) and the result was as expected, the letters were not recognized by the algorithm and it was random. Figure (31) shows this result.

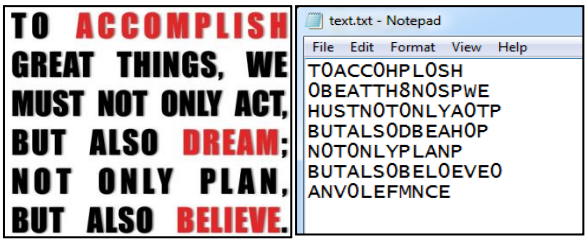

Fig 31: the result without neural network.

5.1.2 If the number of iteration was not high enough

The second theory relating the algorithm is that if the number of iterations was not high enough to approach the correct recognition. Figure (32) shows the result of the recognition of the text from the image in figure (17) with only 400 iterations.

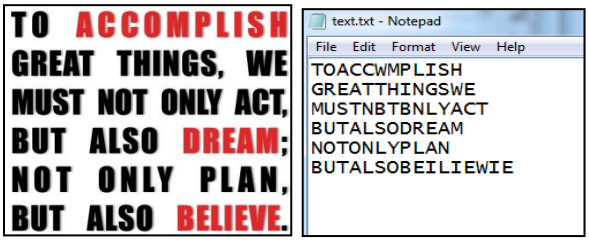

Fig 32: the result if the number of iteration is not high enough

As shown in figure (32) result was not as accurate as the one obtained with 1000 iteration which is shown in figure (18), the algorithm was not able to recognize few letters correctly.

6. CONCLUSION

The purpose of this work is to enhance and optimize the process of recognizing the text in an image. The application of the recognition steps in this paper on several images' types and different situation showed a complete extraction of the letters from the image. Even for strong programs dedicated for converting the text in a scanned document to word file for example; many letters could be missed or not correctly recognized from the image, however, only one or two letters were not extracted correctly in this algorithm, this problem in the proposed algorithm could be easily overcome by training the neural network by more cases. In the future the program could be developed more by establishing a threshold between.

7. REFERENCES

[1] S. Mori, C.Y. Suen, K.Yamamoto, Historical review of OCR research and development, Proc. IEEE, 80(7):1029-1058, 1992.

[2] Y. LeCun, L. Bottou, Y. Bengio, P. Ha®ner, Gradient based learning applied to document recognition, Proc. IEEE, 86(11): 2278-2324, 1998.

[3] C.Y. Suen, K. Kiu, N.W. Strathy, Sorting and recognizing cheques and ¯nancial documents, Document Analysis Systems: Theory and Practice, S.-W. Lee and Y. Nakano (eds.), LNCS 1655, Springer, 1999, pp. 173-187.

[4] Siddhi Sharma1, Neetu Singh2. Optical Character recognition Using Artificial Neural Network Approach. IJETAE. Volume 4, Issue 11, November 2014.

[5] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, Andrew Zisserman. Reading Text in the Wild with Convolution Neural Networks. Springer. International. Journal Computer V. 2016 .116:1-20.

[6] Chen,X.,X & Yuille,A.L.(2004). Detecting and Reading text in natural scenes. In Computer Vision and Pattern Recogntion,2004.CVPR 2004 (Vol.2,pp.II-366). Piscataway,NJ:IEEE.

[7] K. Fukunaga, Introduction to Statistical Pattern Recognition, 2nd edition, Academic Press, 1990.

[8] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, second edition, Wiley Inter science, 2001.

[9] Y. LeCun, L. Bottou, Y. Bengio, P. Ha®ner, Gradient based learning applied to document recognition, Proc.IEEE, 86(11): 2278-2324, 1998.

[10] P.Y. Simard, D. Steinkraus, J.C. Platt, Best practices for convolutional neural networks applied to visual document analysis, Proc. 7th ICDAR, Edinburgh.

[11] Piyush Rai. Kernel Methods and Nonlinear Classification. September 15, 2011

[12] A. Bellili, M. Gilloux, P. Gallinari, An MLP-SVM combination architecture for o²ine handwritten digit recognition: reduction of recognition errors by support vector machines rejection mechanisms, Int. J. Document Analysis and Recognition, 5(4): 244-252, 2003.

[13] A.F.R. Rahman, M.C. Fairhurst, Multiple classifier decision combination strategies for character recognition: a review, Int. J. Document Analysis and Recognition, 5(4): 166-194, 2003.

[14] 'Angel Johncy: Extraction of Connected Component without using BWLABEL in image processing'http://angeljohnsy.blogspot.com/2012/03/extraction-of-connected-components.html. Accessed October 31, 2014.

[15] Third Generation Neural Network. https://www .mql5.com/en/articles/1103.5th5 February 2015.

[16] Quoc V. Le, A Tutorial on Deep Learning Part 2: Autoencoders, Convolutional Neural Networks and Recurrent Neural Networks, October 20, 2015.

[17] Train Stacked Autoencoders for Image Classification.'https://www.mathworks.com/help/nnet amples/training-a-deep-neural-network-for-digit-classification.html' 2016.