Scientific Bulletin of the Petru Maior University of Tirgu Mures
Vol. 6 (XXIII), 2009
ISSN 1841-9267

Handwritten Digits Recognition Using Neural Computing

Călin Enăchescu
"Petru Maior" University of Tirgu Mures
N. Iorga street nr. 1, 540088, Romania
ecalin@upm.ro

Cristian-Dumitru Miron
"Petru Maior" University of Tirgu Mures
N. Iorga street nr. 1, 540088, Romania
cristian_m_17@yahoo.com

Abstract

In this paper we present a method for the recognition of handwritten digits and a practical implementation of this method for real-time recognition. A theoretical framework for the neural networks used to classify the handwritten digits is also presented.

The classification task is performed using a Convolutional Neural Network (CNN). CNN is a special type of multy-layer neural network, being trained with an optimized version of the back-propagation learning algorithm. CNN is designed to recognize visual patterns directly from pixel images with minimal preprocessing, being capable to recognize patterns with extreme variability (such as handwritten characters), and with robustness to distortions and simple geometric transformations.

The main contributions of this paper are related to the original methods for increasing the efficiency of the learning algorithm by preprocessing the images before the learning process and a method for increasing the precision and performance for real-time applications, by removing the non useful information from the background.

By combining these strategies we have obtained an accuracy of 96.76%, using as training set the NIST (National Institute of Standards and Technology) database.

1 Introduction

The ***NIST*** database used to train the neural network [4] contains 60,000 images with distorted handwritten digits, 10,000 images being used for testing and considered as a reference benchmark for applications for handwritten digits recognition.

In order to demonstrate the efficiency of our proposed neural network we will compare the learning performances with a classical ***MLP*** (Multy Layer Perceptron) [5] neural network.

In order to measure the performances of our proposed neural network we will use two different types of neural networks architectures: the first one, a classical ***MLP*** neural network and the second an original ***CNN***, proposed in this paper.

2 The Convolutional Neural Network - *CNN*

CNN is a feed-forward neural network capable to extract topological properties from an image. It extracts features from the input raw image using the first hidden layer and classifies the patterns with the help of the final hidden layer.

The first two layers of the neural network can be considered as a trainable feature extractor [2]. We can add a trainable classifier to the feature extractor, considering 2 fully connected layers (a universal classifier) [2]. The number of hidden neurons contained in the hidden layers is variable and by varying this number we can control the learning capacity and the generalization capacity of the overall classifier [5].

The architecture of the ***CNN*** used to learn the ***NIST*** database is depicted in Figure 1.

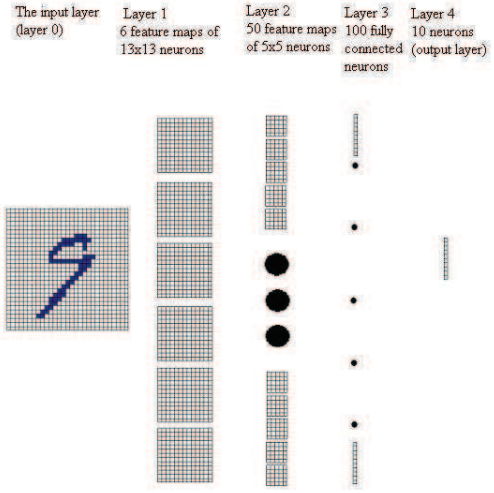

Figure 1: The architecture of the ***CNN***

Keywords: *recognition of handwritten digits, convolutional neural network, learning algorithm*

The general processing strategy for a ***CNN*** is to extract simple features at a higher resolution, and then to convert them into complex features at a coarser resolution. Each convolutional hidden layer can reduce the resolution of the initial image by a factor of (n-3)/2. For this purpose, each neuron contained in a convolutional hidden layer will process a single region from the map of the previous layer. In this way a neuron is functioning like an independent micro kernel, named ***convolutional kernel*** [2]. The convolutional kernel contains as inputs the corresponding region from the previous map to be processed.

The width of the convolutional kernel is chosen to be centered on a unit-pixel (odd size), to have sufficient overlap for not losing information (3 units are too small with only one unit overlap, 7 units represents a 70% overlap). A ***convolution kernel*** of size 5 is shown in Figure 2.

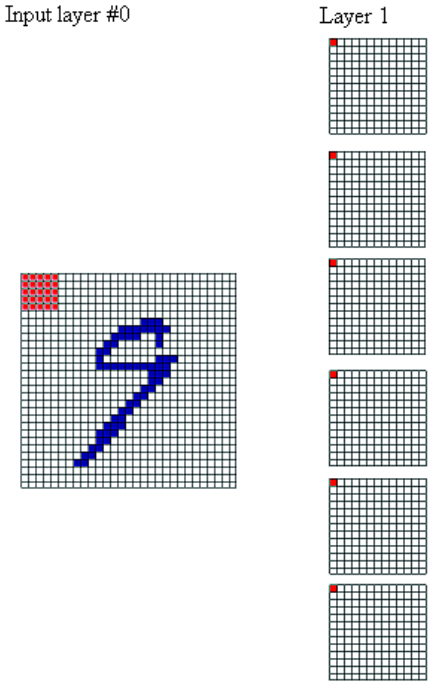

Figure 2: The inputs of the first neuron from each feature map in Layer1

With no padding, a sub-sampling [2] of 2, and a kernel size of 5, each convolution layer reduces the feature size from n to (n-3)/2. Since the initial ***NIST*** database contains images of size 28x28, the nearest value which generates an integer size after 2 layers of convolution is 29x29.

The ***CNN*** architecture is composed of 4 layers. The first layer is a convolutional layer composed of 6 feature maps of 13x13 units. Each neuron processes only a region of 5x5 pixels from the input image, ignoring the rest of the information from the picture. The second layer is also a convolution layer that is processing a region of 5x5 outputs from each feature map of the previous layer, the other outputs being ignored. The third layer is composed of 100 neurons fully connected with the neurons from the second layer. The output layer is composed of 10 neurons, one neuron for each pattern that has to be classified.

A trainable weight is assigned to each connection, but all units of one feature map share the same weights. This characteristic is justified by the fact that an elementary feature detector useful on one part of the image is unlikely to be useful for the entire image. Moreover, the weight sharing technique allows the reduction of the numbers of trainable parameters. For instance, ***CNN*** has only 132,750 trainable parameters out of 303,450 connections.

3 Cutting out useless information

Although ***CNN*** are known to be able to extract information from images with minimum preprocessing, the removal of the un-useful pixels from the background improves performances, not only during the learning process, but more important, during the use of the ***CNN*** in practice. One of the reasons is that we have the ***NIST*** training database that contains 60,000 images of handwritten digits but, all of them are centered and have the same width. In practice we will have to extract out digits from bigger images and resize them to 29x29 pixels, so we can present them as inputs to the neural network. Trying to cut out a digit from an image, resizing and centering the image, we can obtain the same results as with the digits contained in the ***NIST*** database. Another advantage of this strategy is obtained when an image of the digit is zoomed in and the relevant patterns are accentuated. In Figure 3 we have on the left a sample image from the ***NIST*** database and on the right the same image with the background removed.

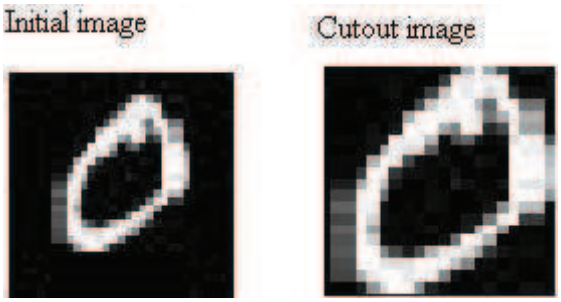

Figure 3: Background removal

4 Pre-encoding of the training images

The encoding of the images used in the learning process is a crucial improvement of the learning process because it dramatically decreases the time needed to pass one learning epoch [5]. The images are codified using the following rule: the useful information, the white pixels, are set to 1 and the background pixels are set to 0.

After this encoding process we will store the encoding images in a special training file. Before starting the learning process we will load all this samples from the training file into the internal memory of the computer. It is very important to run the learning process with the deactivated virtual memory option of the operation system - this will eliminate the need to use the external memory which is slow compared with the internal memory. An example of encoding is presented in Figure 4.

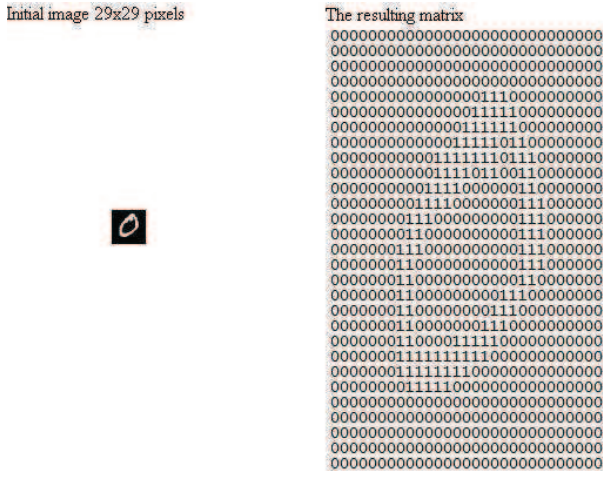

Figure 4: Encoding example for digit 0

5 The learning process

The learning process was based on the ***NIST*** training database that contains 60,000 images of handwritten digits. Because the training set is huge, the duration of a single learning epoch took almost 73 minutes on a computer with Intel Dual Core processor - 1.86 GHz and with 3 GB of RAM. Using our strategy presented previously, firstly we have pre-encoded the training images and then we have loaded them into the system internal memory, one time, at the beginning of the learning process. In this way we have reduced the time span for one epoch to 52 minutes, that is an improvement of 40% in performance.

An important parameter that was affecting the learning process is the learning rate [5]. A high learning rate was causing a bad learning performance. Using small learning rates the ***CNN*** had good learning performances. An example of an efficient learning rate is 0.01. To obtain good results, in order to have a constant error decrease, we have used different learning rates in different stages of the learning process: the learning rate was reduces with 10% at every 3 epochs. Another important thing was to use very small initial values for the synaptic weights - we have obtained the best results using initial values for the weights in the interval [-0.005, 0.005].

We have used the hyperbolic tangent [5] as activation function. The graph of the learning process is presented in Figure 5.

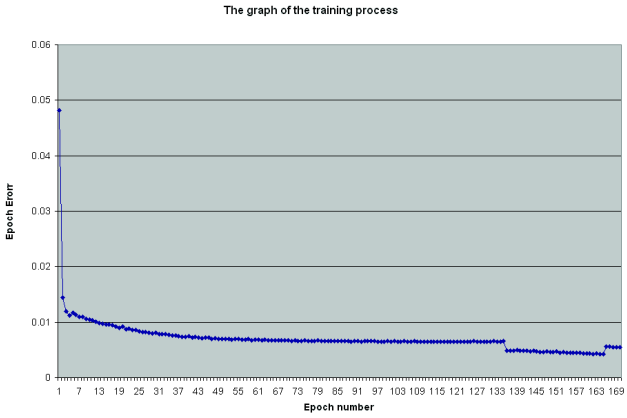

Figure 5: The graph of the learning process

6 The testing process

After finishing the training process, the neural network was tested in 2 different ways: using the ***NIST*** testing database, containing 10,000 images of handwritten digits that weren't presented in the learning process and by testing the ***CNN*** capacity to recognizing handwritten digits in real time.

In the first testing we have pre-encoded the testing images to obtain better performance, and to save time. The neural network obtained the following performances: with the original ***NIST*** testing images **96.74%** accuracy and **96.56%** accuracy with the images without background.

Table 1 contains the best results obtained so far in the world, they range from an error of 12% up to an error of 0.4%, using the ***NIST*** testing database with10,000 images.

Table 1: Results of the learning process for different neural networks

Method	Error	References
Linear classifier	12%	[1]
Linear classifier (nearest neighbor- NN)	8.4%	[1]
Pair wise linear classifier	7.6%	[1]
K-NN, Euclidean	5.0%	[1]
2 layer NN 300 hidden units	4.7%	[1]
2 layer NN 1000 hidden units	4.5%	[1]
2 layer NN 1000 hidden units using distortions	3.6%	[1]
1000 RBF + linear classifier	3.6%	[1]
Our CNN	3.26%	this paper
LeNet-5	0,8%	[3]
Convolutional NN elastic distortions	0.4%	[2]

7 Conclusions

This article presents a method for analysis of visual documents and can be considered a starting point for the problem of handwritten letters recognition and handwriting recognition. In the future we will try to extend this architecture to support the recognition of handwritten letters based on the ***NIST*** database of handwritten letters containing 800,000 images. New methods of training the network must be found to speed up the learning process and to manage this huge learning set.

References

[1] E. Kussul, T. Baidyk, "Improved method of handwritten digits recognition. UNAM", Centro de Instrumentos, Cd. Universitaria A.P. 70-186 , CP 04510, Mexico D.F, 2002.

[2] P.Y. Simard, D. Steinkraus, J.C. Platt, "Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis", Microsoft Research, One Microsoft Way, Redmond WA 98052, 2003.

[3] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, "Gradient-based learning applied to document recognition", Proceedings of the IEEE, v. 86, pp 2278-2324, 1998.

[4] http://www.cs.toronto.edu/ roweis/data.html

[5] C. Enăchescu, "Calculul neuronal", Casa Cărţii de Ştiinţă, ISBN: 978-973-133-460-8, Cluj Napoca, 2009.