BULETINUL Universităţii Petrol – Gaze din Ploieşti	Vol. LXI No. 3/2009	365 - 370	Seria Tehnică

Using Artificial Neural Networks in a Pattern Recognition Control System

Daniela Şchiopu

Petroleum – Gas University of Ploieşti, 39 Bucureşti Blvd., Ploieşti, ROMÂNIA
e-mail: daniela_schiopu@yahoo.com

Abstract

Artificial neural networks present a current research topic leading to a range of disciplines, including computer science (artificial intelligence) and have applications in industry, medicine, biology or physics. This paper discusses a system for pattern recognition which uses artificial neural networks, system that recognizes and classifies pieces in a manufacturing process. The system also recognizes out of order pieces for exclusion them from the process. Implementation was made in Matlab® 7.1 using the Neural Network Toolbox and Image Processing Toolbox.

Key words: *artificial neural networks, pattern recognition, training algorithm.*

Introduction

Neural networks are special units of information processing: they consist of many primitive cells (artificial neurons) working in parallel and are connected by direct links (connections).

The main method used by these cells refers to the distribution of patterns formed by the activation (output) of the cell groups, along the connections, similar to the basic mechanism of human brain, where information is transfered from a group of neurons to others through synapses. The modelling of this mechanism of human brain functioning by artificial neural networks was motivated by the high performances of the human brain in complex cognitive tasks such as pattern recognition.

Using these important properties of neural networks, the author presents the following case: a robot should recognize certain pieces of a manufacturing process (screw, washer) to classify the good pieces and exclude out of order pieces from the process.

The robot is equipped with a camera, and each piece of the conveyor belt is held in the memory of the robot as an image in *png* format and is the form the robot needs to recognize later. The system has a training data set, which contains both images of the correct pieces, and the defective ones. The training data set includes images taken over certain sites that sell pieces, and we modified certain pixels in some of them for obtaining "defective" images. Neural network learns on this training set, so that the network makes a more accurate classification to a new capture (a picture of a piece). The system can be successfully implemented in a real situation. This paper is based on the results obtained by the author in her master thesis, presented in March 2009.

Artificial neural networks concept

The artificial neural network was first introduced by Kohonen (1982) as the "self-organizing feature map" [1, 2, 7].

The pattern recognition systems required large numbers of vast and complicated algorithms, because the differences between machines and human beings [3, 6]. For example, the way people have acquired the ability to recognize pictures can only be by experience. By trial and error, they have learned to perform certain tasks. Machines are preprogrammed and if they can learn, they are restricted to certain classes of preprogrammed methods of learning.

This comparison between the behavior of artificial machines and the behavior of human brain led to using neural networks model for developing more intelligent machines.

Functionally, an artificial neural network is a system which receives input data (corresponding to initial data of a problem) and produces output data (which can be interpreted as responses of the problem considered). A key feature of neural networks is the ability to self-adapt to the informational environment, corresponding to a concrete problem through a learning process. Thus, the network extracts the problem model from examples.

From the structural point of view, a neural network is a set of interconnected units (neurons) each unit been characterized by a simple operation. The units' actions are influenced by a number of adaptive parameters. Thus, a neural network is an extremely flexible system.

Therefore, a neural network is a massive parallel and distributed processor, which has a natural tendency to stockpile experimental knowledge and make it available for use. It resembles the human brain in two ways:

- Knowledge is obtained by the network through a learning process;
- Knowledge is not stored in the processing units (neurons), but inter-neural connections, known as synaptic weights.

Artificial model of a neuron

Artificial neuron is a simplified model of an organic neuron, for which one can distinguish three different parts: a set of incoming fibers (the dendrites), a cell body (the soma) and one outgoing fiber (the axon).

In the case of artificial neuron, the structure is very simple. It has: a set of input connections, a set of output connections, a level of activation, an output value, a residual value, a function of activation (Fig. 1).

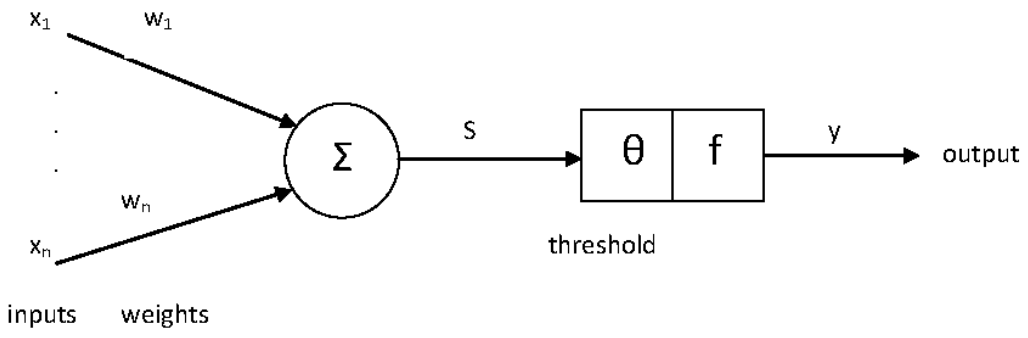

Fig. 1. Neuron structure.

Each connection has a real value, called synaptic weight, which determines the effect that the entry level of neuron activation. In Fig. 1, x_i represents inputs; w_i – synaptic weights, f – the activation function, θ – the threshold, and output y, which is calculated by:

$$y = f\left(\sum_{i=1}^{n} w_i x_i + \theta\right) \tag{1}$$

Neural networks used in pattern recognition

Neurons may be connected in various ways to form a neural network. A typical model of topology considers neurons organized in multiple layers.

A multilayer neural network contains two or more layers of neurons. The first layer receives inputs from the environment. The neurons outputs of this layer are the input for next layer neurons. The output of the entire network consists of the last layer neurons outputs. Layers between the first and the last layer are the hidden layers of the network. Nodes in the input layer are only intended to send the inputs, not computational role and, therefore, they are not deemed to form a layer.

To increase performance and solve more complex problems we can create feedforward networks with multiple layers. In these networks, output of a certain level is entry in immediate upper level. The levels between input pseudo-level and output level are called hidden levels.

Backpropagation neural networks have an input layer, an output layer and possibly one or more hidden layers. Links between nodes of different layers have certain values which are determined by iterative process of learning.

Backpropagation algorithm

The most important stage in the design and implementation of artificial neural networks of any type is learning. The network performance depends largely on success of this phase.

In the context of neural networks, learning is a process whereby the variable parameters of a neural network are adapted through a continuous stimulation process from the environment in which it is included. Type of learning is determined by how parameters change.

Backpropagation algorithm is the most known and used supervised learning algorithm. Called also generalized delta algorithm, it is based on minimizing the difference between desired output and actual, known output, using descending gradient method. The method was first proposed by Bryson and Ho (1969), but at that time it was ignored because it asked too many calculations. It was then rediscovered by Werbos (1974), but was launched just in the mid-'80 by Rumelhart, Hinton and Williams (1986) as generally accepted tool for training of multilayer perceptron [2]. The basic idea is to find the minimum of error function in relation to the weights of connections.

The most common form of this algorithm is [2, 5]: the network is first initialized by setting up all its weights to be small random numbers. Next, the input pattern is applied and the output calculated (the forward pass). The calculation gives an output which is different to what we want (because the weights are random). Then we calculate the error of each neuron (being what we want – what we actually get). This error is then used mathematically to change the weights in such a way that the error will get smaller (the backward pass). The process is repeated until the error is minimal or all examples are classified correctly or stopping criterion is satisfied.

Application of neural networks in pattern recognition

The human ability to recognize, apparently without much effort, different pictures as equivalent, challenges scientists to copy the neuro-physiological mechanisms involved in human pattern recognition. For this reason, the processing of information by artificial neural networks has gained a lot of interest of many scientists in recent years.

In this paper, we consider the following case. Suppose we use a robot having a camera (transducer) to observe a certain object coming on a conveyor belt in a production line. The robot saves the captured picture, and the system software takes this picture as input and gives response accordingly. Finally, the robot must be able to identify and classify that piece.

The recognition and classification software is implemented in Matlab® 7.1 and uses – in addition to its basic operating with matrix functions and specific image processing functions – neural networks. These extensions of Matlab® are Image Processing Toolbox and Neural Network Toolbox [4].

The neural network achieved is a totally connected feedforward network. Neural network consists of three layers: an input layer, an output layer and a hidden layer. Input layer has 744 neurons (training images and test images are "png" files with size 24 x 31 pixels, so 744 pixels total). Output layer has two neurons, one for each of the two classes of recognition: class "screw" and class "washer". Hidden layer has 100 neurons. The number of neurons in hidden layer is chosen from the experience of the network designer. If problems appear in the training of the network, then neurons will be added on hidden layer. Consequently, the network is a 744 / 100 / 2 net (see Fig. 2).

The activation functions are *tansig* (hyperbolic tangent function) for hidden layer, and *logsig* (sigmoid function) for output layer.

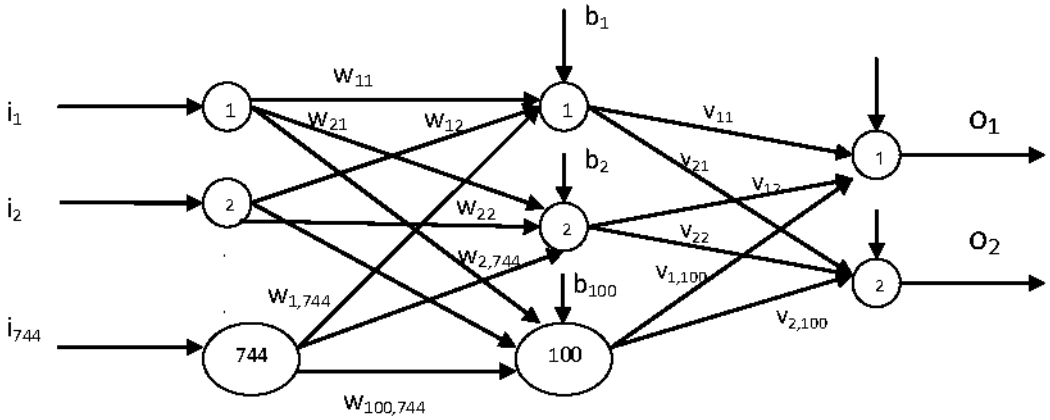

Fig. 2 Neural network architecture.

The network is trained with 45 training images; the filenames are encoded so that when the application loads image training, the program will know the desired network output.

The outputs were coded with 00 for class washer, 11 for the class screw, 01 for defective screw and 10 for defective washer.

The network receives *nr* input Boolean values, where *nr* is the number of files in the folder that reads training images – in our case *nr* is 45.

As output, we have a matrix *t* with 2 rows and *nr* columns, which encodes the desired outputs for each case.

Polarizations and weights of the network are randomly initialized.

For training neural networks we can use *adapt* or *train*. Function *train* applies each vector of the training set and computes variations of the weights and polarizations for each input [5]. Then the network is updated with the sum of these corrections. Each pass over all input vectors is called the epoch. In the *adapt* function, the weights are adjusted after each input vector.

The application has a GUI (Graphic User Interface) that is available to the Matlab® user. In the graphical interface are used different parts of Matlab® GUI option: uicontrol objects (buttons), uipanel objects, axes objects.

The first button, "Imagini de antrenament", aims to show all images in specified folder (max. 45) in the panel on the left, to be included in the training stage of the network.

These images go through a transformation process (the button "Prelucrare imagini"). First the image is transformed into a grayscale image, using the *rgb2gray* function, then in a black and white image with the *dither* function. Each matrix obtained is then resized, transformed into a vector with 1 row and 744 columns, using the *reshape* function. The button "Creeaza reteaua" creates and trains the neural network.

The button "Imagine de test" allows the user to upload a file to test the network created and trained.

After the test image is loaded (it will be viewed after its conversion into black and white image), for its recognition and classification in one of the four classes by using the "Recunoaste" button.

The result for a given loaded image is (Fig. 3):

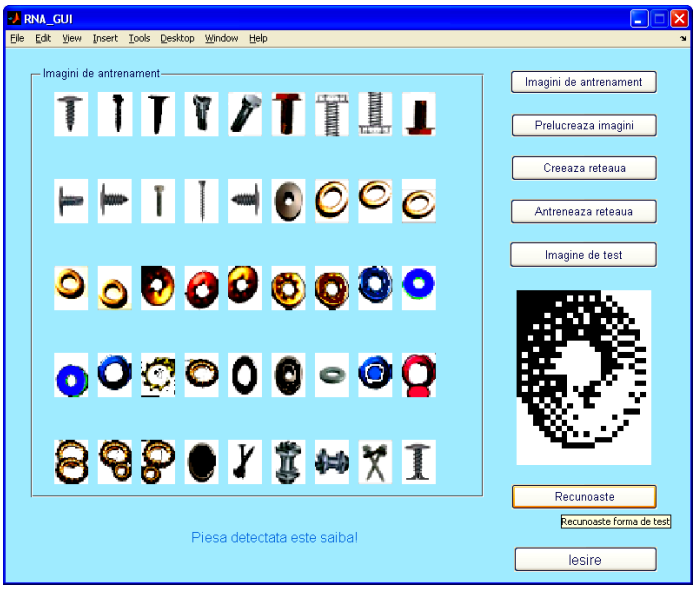

Fig. 3 The Matlab® GUI for test image recognition and display of an appropriate message.

The main purpose of the system is to obtain information from images, which can be considered two-dimensional matrix, or even three dimensions matrix and which are actually projections of the environment on the system. This information refers to the recognition and classification of input data into a number of four classes.

Thanks to the libraries of functions provided by Matlab® to solve this problem, and combining these benefits with the algorithms and theoretical notions about artificial neural networks, it was possible the application presented in this paper.

Conclusions

The basic properties of the neural networks – namely that they are universal approximated and they have a special prediction capacity – find immediate use in some areas, such as image processing, visual pattern recognition or speech recognition .

The pieces recognition system, which we implemented in this paper, combines the advantages of artificial neural networks with those of Matlab® environment and may even have applicability in a manufacturing process of different types of pieces.

As further developments, the application can be improved by using – as training data and test data – some images with other dimensions than those used in this paper (24 x 31 pixels); using other types of neural networks and the inclusion of other classes of recognition in implementation (in the application are actually only two classes – "screw" and "washer"), or using other training methods for neural network. Also, we can use test images with other dimensions than those of training images for pattern recognition.

References

1. Căleanu, C.D., Tiponuţ, V. – *Reţele neuronale. Aplicaţii.* Ed. Politehnica, Timişoara, 2002.
2. Căleanu, C.D., Tiponuţ, V. – *Reţele neuronale. Arhitecturi şi algoritmi*. Ed. Politehnica, Timişoara, 2001.
3. Jain, A., Duin, R., Mao J. – *Statistical Pattern Recognition: A Review.* IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol. 22, No. 1, January 2000.
4. *** – *Matlab 7.1. Neural Network Toolbox. Image Processing Toolbox*. Mathworks Inc., 2005 (http://www.mathworks.com/).
5. Moise, A. – *Reţele neuronale pentru recunoaşterea formelor*. Course notes, 2008.
6. Ripley, B.D. – *Pattern Recognition and Neural Networks.* Cambridge University Press, 1996.
7. Veelenturf, L.P.J. – *Analysis and Applications of Artificial Neural Networks.* Prentice Hall, 1995.

Utilizarea reţelelor neuronale artificiale într-un sistem de recunoaştere a formelor

Rezumat

Reţelele neuronale artificiale reprezintă la ora actuală un domeniu de cercetare de vârf pentru o serie de discipline, printre care ştiinţa computerelor (inteligenţă artificială) şi au aplicaţii în industrie, medicină, biologie sau fizică. Acest articol prezintă un sistem de recunoaştere a formelor bazat pe reţele neuronale artificiale, care recunoaşte şi clasifică piese în cadrul unui proces de fabricaţie. Sistemul recunoaşte, totodată, şi piesele defecte în vederea excluderii lor din proces. Implementarea s-a făcut în Matlab® 7.1, folosindu-se Neural Network Toolbox şi Image Processing Toolbox.