THE CLASSIFICATION OF FACIAL EXPRESSIONS USING THE LEVENBERG-MARQUARDT BACKPROPAGATION ALGORITHM

Gîlcă Gheorghe, *"Constantin Brîncuși"* University from Tîrgu-Jiu, ROMANIA
Gîlcă Natalia, *Roșia de Amaradia* Technological High School, Gorj, ROMANIA

ABSTRACT: This article aims to implement a facial recognition system based on the feedforward neural networks. The used neuronal network is instructed using the Levenberg Marquardt backpropagation algorithm. This algorithm performs well in a short time requires reduced memory and can recognize the 7 states of basic emotions. The topologic structure feedforward network comprises: the input level, the hidden level and the output level. The neural network is based on the values of the three facial components, which most varies from one emotional state to another: the mouth, eyebrows and eyelids. The output system allows the recognition of the emotional states (the 7 base states) containing one neuron for each class of emotions.

KEYWORDS: facial expressions, neural networks, backpropagation, emotion.

1. INTRODUCTION

In the scientific literature on artificial neural networks (ANNs) great progress was made in understanding the principles of how they work, which corroborated with the rapid growth of the available computing resources led to the development of a large number of modeling, identification, prediction and control applications based on ANNs [1] - [3], especially in mechanical engineering and robotics. The ability of neural networks to approximate complex nonlinear relationships without prior knowledge of the model structure makes them a very attractive alternative to classic techniques of modeling and control. Most applications are implemented with feedforward neural networks (FFNN) [4] - [6], this classic model based on Nonlinear AutoRegressive Moving Average (NARMA).

Feedforward neural networks generally have a static structure therefore, they are appropriate to approximate relationships, mainly static (non-linear). FFNN applications in real-time for dynamic systems require the introduction of external delay feedback [5]. Recurrent neural networks (RNN) has its own internal delayed reactions in time, so they are promising alternatives to identify and control the system, especially when the task is to model dynamic systems [7].
The aim this paper is to apply in order to train the RTNN network the Levenberg Marquardt Backpropagation control algorithm(L-M) and the result will be identifying and classifying human facial expressions for people.

2. THE RTNN NETWORK TOPOLOGY

A recurrent trainable neural network model (RTNN) and its learning algorithm of dynamic Backpropagation type, with explanatory diagrams and stability tests are described in [8]. RTNN topology, is expressed in the vector matrix form and is described by the following equations:

$$X(k+1) = AX(k) + BU(K) \quad \textbf{(1)}$$

$$Z(k) = S[X(k)] \quad \textbf{(2)}$$

$$Z(k) = S[CZ(k)] \quad \textbf{(3)}$$

$$A = block - diag(A_{ii});\ |A_{ii}| < 1 \quad \textbf{(4)}$$

Where Y is the output, X is the state vector, U is the input vector with the dimensions l, m, n; A is a diagonal weighted matrix of states having nxn size; A_{ii} is the i-th element of the diagonal matrix A and has the size (lxl). Equation (4) represents local stability conditions, imposed on all elements of the matrix A; B and C are input, respectively output, weighted matrix , and they have the sizes (nxm) and (LXn). S is a tangent activation hyperbolic function or a sigmoid with vector values, which allows the identification of the RNN network, and k is the discrete time variable.

3. TRAINING THE RTNN NETWORK USING THE LEVENBERG MARQUARDT RECURSIVE ALGORITHM

Generally the recurrence backpropagation update rule is described in [8] and is defined as:

$$W_{ij}(k+1) = W_{ij}(k) + \eta\Delta W_{ij}(k) + \alpha\Delta W_{ij}(k-1) \quad \textbf{(5)}$$

Where W_{ij} is a general weight, ΔW_{ij} is the weight correction for the overall weighted matrix, while η and α are parameters of the learning rate. The updates of the weight are calculated with the following formula:

$$\Delta C_{ij}(k) = [T_j(k) - Y_j(k)]S_j^{'}[Y_j(k)]Z_i(k) \quad \textbf{(6)}$$

$$\Delta A_{ij}(k) = RX_i(k-1) \quad \textbf{(7)}$$

$$R = C_i(k)[T(k) - Y(k)]S_j^{'}[Z_i(k)] \quad \textbf{(8)}$$

$$\Delta B_{ij}(k) = RU_i(k) \quad \textbf{(9)}$$

Where $\Delta A_{ij}(k)$, $\Delta B_{ij}(k)$, $\Delta C_{ij}(k)$ are the weight correction for each matrix (A_{ij}, B_{ij}, C_{ij}); (T-Y) is the error vector of the output layer of the RTNN network, T is the desired target vector and Y is the RTNN network output vector; R is an auxiliary variable, $S_j^{'}$ is the derived of the activation function.

The Levenberg Marquardt trecursive algorithm for training the RTNN network is defined by the following equations [9] - [11]:

$$W(k+1) = W(k) + P(k)\Delta Y[W(k)]e[W(k)] \quad \textbf{(10)}$$

$$Y[W(k)] = g[W(k), U(k)] \quad \textbf{(11)}$$

$$E[W(k)] = e^2[W(k)] = \{g[W(k), U(k)] - Y_p(k)\}^2 \quad \textbf{(12)}$$

$$\Delta Y[W(k)] = \frac{\partial}{\partial W} g[W, U(k)] \quad \textbf{(13)}$$

4. THE RESULTS OF THE SIMULATION

We used in the simulation Matlab R2012b installed on a PC with the configuration: Intel Celeron Dual-Core T3500 and 4GB RAM memory. The feedforward network structure is shown in figure 1 and is composed as follows: input level, consists of three input vectors plus the 7 outputs of the network, the intermediate level (hidden), consists of five neurons and the output layer comprises 7 neurons, one for each emotional state.

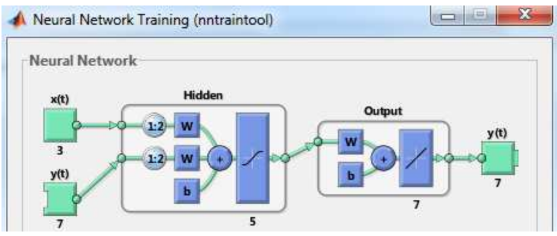

Figure 1. The feedforward network structure

The neural network used is built using the Levenberg Marquardt backpropagation algorithm. The 3 input vectors have 41 different values, being called feature vectors (3x41, resulting in 123 elements), with 123 neurons associated . The output vector is of the 7x1 form, resulting 7 elements corresponding to the 7 classes: happiness, sadness, fear, anger, surprise, disgust and neutral.

The mean squared error graph is presented in figure 2. The best validation performance of the network is 0.025602 obtained at epoch 68.

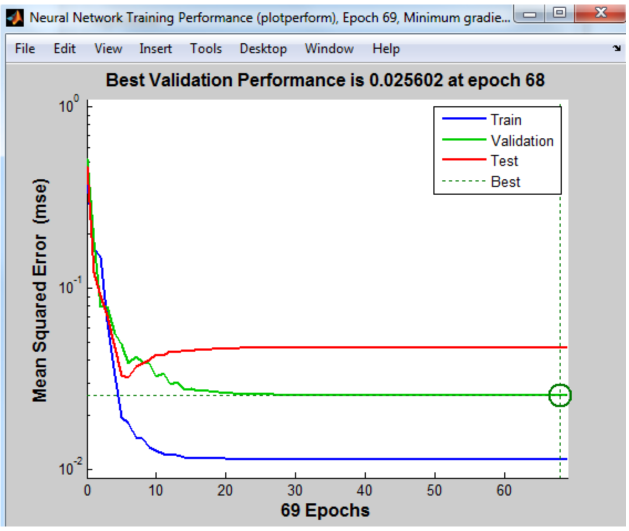

Figure 2. The display of Mean Squared Error

Figure 3 shows the response of trained network. Here are displayed processes: training, validation, and testing schedule for network error. The network has trained for most instances of error values approximately equal to 0, only 5 instances are values greater than 0.

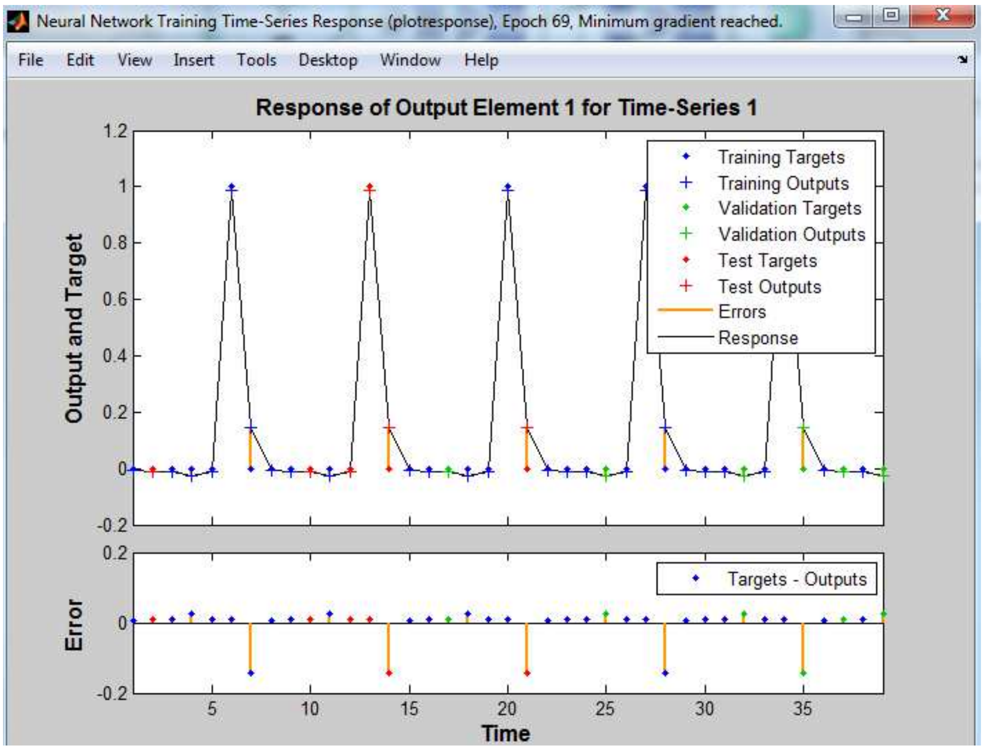

Figure 3. The response of the network in time

Figure 4 shows the error autocorrelation function. For an ideal predictive model only one error value should be non-zero,aligned with 0 on the axis labeled La. In this case we have four lines that have nonzero values and they exceed reliable limits.

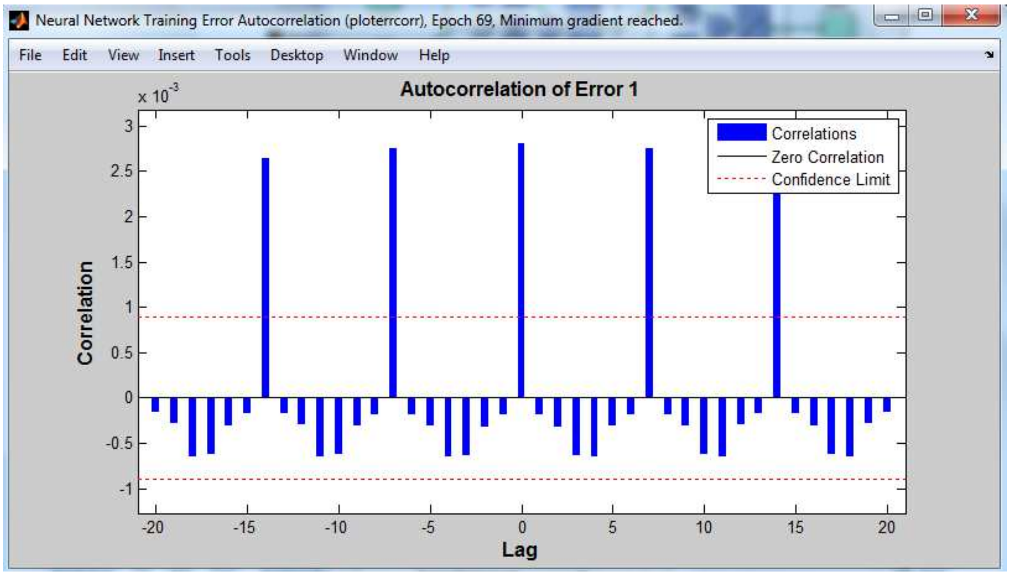

Figure 4. The display of the autocorrelation function of error

Figure 5 shows the cross-correlation input-error functions. These functions illustrate how errors are correlated with the input sequence. For a perfect prediction model, all correlation should be zero. In our case, all correlations fall within the reliability limits around the value zero, marked with dotted red lines.

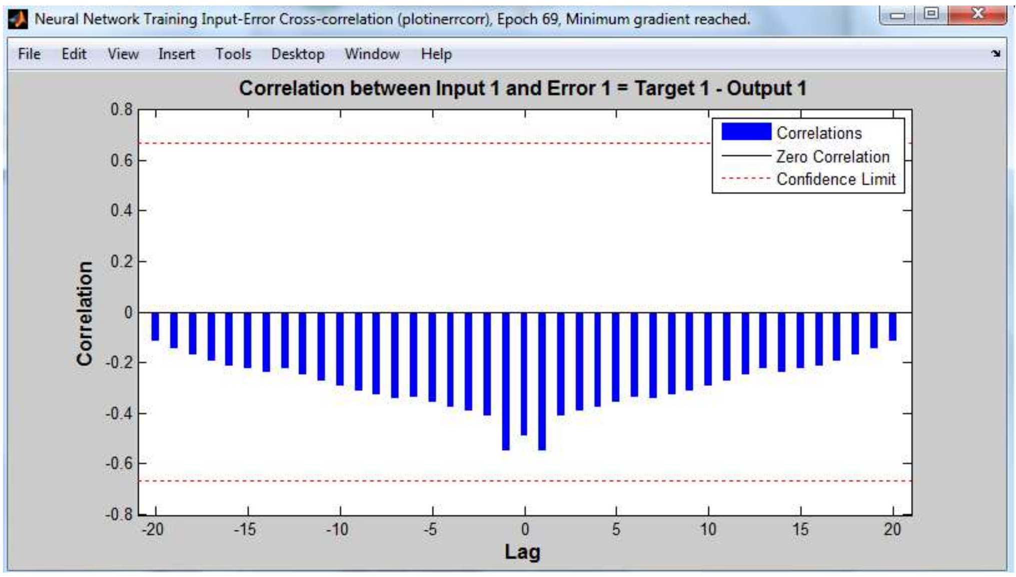

Figure 5. The cross-correlation input-error functions.

Figure 6 shows the error histogram. For most instances (a total of 150) the error is almost zero, only for about 5 instances is higher, resulting that the classification system is optimal in the classification of emotion classes.

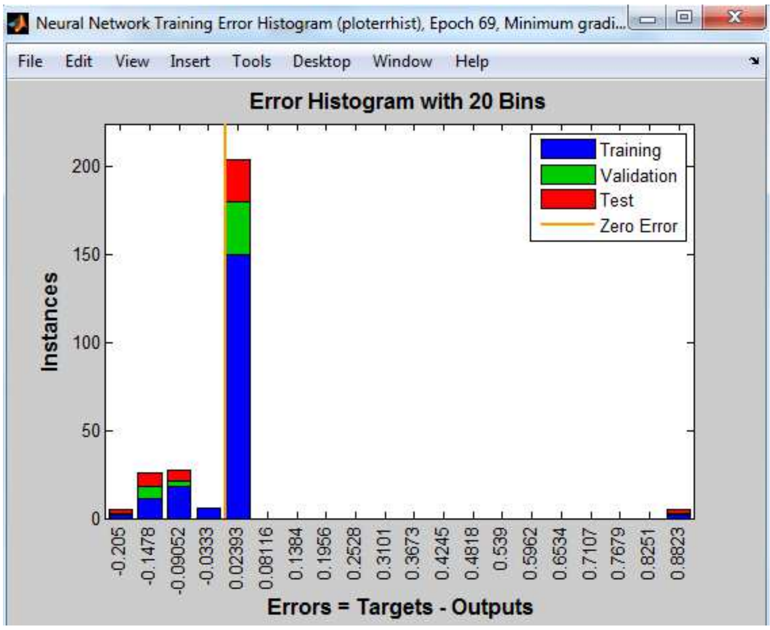

Figure 6. The error histogram

Figure 7 shows the regression model generated by the network for the training processes , validation, testing and all of these mixed. The regression model determines the belonging of the network's input points to a class of emotions from its output. From the training process of the neural network we can see that its output is approximately equal to the set target, the conclusion being that the used network optimally classifies the input data.

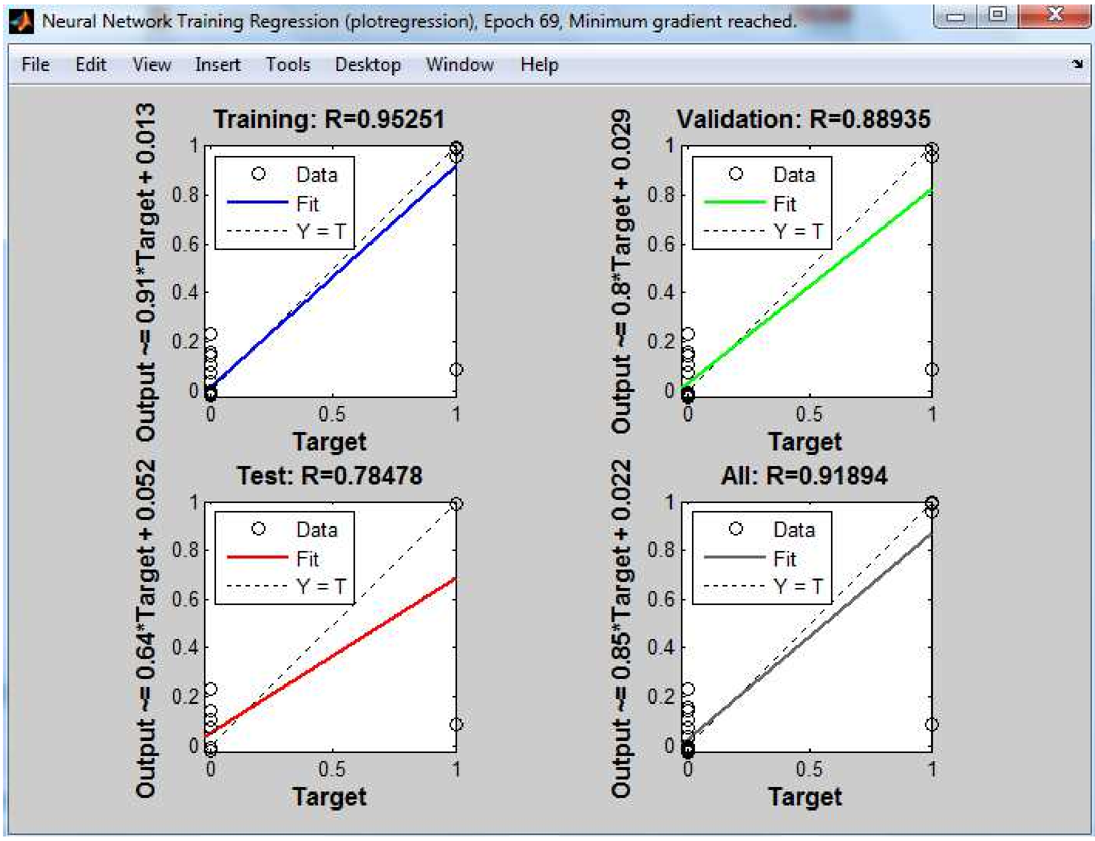

Figure 7. The regression model generated by the artificial neural networks

5. CONCLUSION

This paper shows a human emotions recognition system based on the feedforward neural network trained with the Levenberg-Marquardt backpropagation algorithm.

The structure of the network consists of three levels: input with 123 neurons, the hidden layer with 5 neurons and output level with seven neurons.

For the input level the network uses data for three vectors corresponding to facial features that change quickly for each state, namely: eyebrow, eyelid and mouth, and for the output level we have the emotional states: happiness, sadness, fear, anger, surprise, disgust and neutral.

The system can classify without problems 7 emotion classes without confusing them (happiness, anger, fear, anger, surprise, disgust and neutral), given that the artificial neural network output is approximately equal to the target set.

BIBLIOGRAPHY

[1]Brugge M, Stevens J, Nijhuis J, Spaanenburg J. (1998). License Plate Recognition using DTCNNs. Fifth IEEE International Workshop on Cellular Neural Networks and their Applications.

[2]Chang S, Chen L, Chung Y, Chen S. (2004). Automatic License Plate Recognition. IEEE Transactions on Intelligent Transportations Systems.

[3]Lim B, Yeo W, Tan K, Teo C. (1998). A Novel DSP Based Real-Time Character Classification and Recognition Algorithm for Car Plate Detection and Recognition. Proceedings of ICSP.

[4]Lin-Liu Ch, Nakashima K, Fujisawa H. (2003). Handwritten Digit Recognition: Benchmarking of State of the Art Techniques. Pattern Recognition.

[5]Nijhuis J, Brugge M, Helmholt K, Plium J, Spaanenburg L. (1995). Car License Plate Recognition with Neural Networks and Fuzzy Logic. Proceedings, IEEE international Conference on Neural Networks.

[6]Pan X, Ye X, Zhang S. (2005). A Hybrid Method for Robust car Plate Character Recognition. Engineering Applications of Artificial Intelligence.

[7]Park S H, Kim K I, Jung K, Kim H J. (1999). Locating Car License Plates using Neural Networks. Electronics Letters Online.

[8]Ryung E, Pyeoung K, Joon H. (1994). Automatic recognition of a car license plate using color image processing. IEEE International Conference.

[9]Sirithinaphong T, Chamongthai K. (1999). The recognition of car license plate for automatic parking system. Fifth International Symposium on Signal Processing and its Applications.

[10] Zheng D, Zhao Y, Wang J. (2004). An Efficient Method on License Plate Location. Pattern Recognition Letters.

[11] Zimic N, Fickzo J, Mraz M, Virant J.: The Fuzzy Logic Approach to the Car Number Plate Locating Problem. Intelligent Information Systems, Proceedings.