Comparison of Parkinson's Disease Diagnosis using SOM and MLP Neural Networks

Camilo A. Torres[1], Diego A. Mora[2], Orjuela-Canon, A.D.[3]
Department of Electronic Engineering
Universidad Distrital Francisco José de Caldas
Bogotá, Colombia

***Abstract*— This paper realizes a comparison of two methods used as solution to the classification problem of patient's with Parkinson´s Disease from measures taken on bosses of voice. The used methods are Multilayer Perceptron (MLP) and Kohonen's self-organizing maps (SOM). Both methods are addressed with and without data preprocessing using Principal Components Analysis (PCA). The best classification result obtained is with MLP without data preprocessing, which achieved a high correct average classification rate of 90.24%, therefore it can be taken into account for the Parkinson's Disease diagnosis. Finally a comparison between the addressed solutions is made in terms of correct classification rate.**

Keywords*— *diagnosis, Multilayer Perceptrons, Parkinson´s Disease, PCA, Self Organizing Maps, supervised learning, unsupervised learning.

I. INTRODUCTION

Parkinson's is a degenerative disease that generates movement disorders, altering coordination, reflex, posture, sleeps disorders, loss of sense of smell, depression and cognitive impairments among others. Since the symptoms of this disease can be attributed to other disorders, diagnosis is complex, especially in the early stages of the disease. The diagnosis method addressed in this paper is the measurements analysis of recorded speech signals from 32 people, 9 with Parkinson´s Disease [1].

Among the methods that have been addressed to solve this problem are: Support Vector Machine [2] with the same dataset that is used in this work (recorded speech signals), achieving a correct average classification rate of 91.4%; Self Organized Neural Nets [3], using the Hoehn and Yahr classification with a correct classification rate of approximately 90%; Self Organized Maps, analyzing movement coordination trough drawings with an optical mouse [4] and Fuzzy Expert Systems using recorded speech signals of the PhysioNet database [5].

In the present work, the Multilayer Perceptron (MLP) and Kohonen´s Self-Organizing Maps (SOM) are used to carry out the classification. The principal component analysis (PCA) is performed in both methods and the results are compared with the diagnosis with the complete dataset. Finally, a comparison between MLP and SOM is performed.

The paper is organized as follows: Section II explains the experimental methodology. Section III explains the data preprocessing trough Principal Components Analysis (PCA) method. Section IV presents the classification using Multilayer Perceptron with and without applying PCA. Section V presents the classification using a Kononen's self-organizing map with and without applying PCA. Section VI make a comparison between the obtained results using MLP and SOM and finally the section VII presents the conclusion of this work.

1. caatorresc@correo.udistrital.edu.co
2. damorag@correo.udistrital.edu.co
3. dorjuela@ieee.org

II. METHODOLOGY

The experimentation is carried out in the following way: For both methods (MLP and SOM), there are performed two experiments: with and without applying preprocessing of data, and for each experiment several variants are realized; these are change the optimization function, neurons in the hidden layer, and learning rate for MLP; and change map dimensions, neighborhood function and neighborhood topology for SOM. The obtained results are compared in terms of the average correct classification rate which determine the ability to generalize of the neural network.

III. DATA PREPROCESSING

The original dataset is composed by 192 instances and 22 attributes (measures extracted from voice recordings), distributed in 6 samples taken at 32 patients. To reduce neural network complexity, a transformation of the original dataset is realized with the aim of decrease the number of attributes taken as much information as possible contained in the original dataset.

The Principal Components Analysis [7] is based on the data correlation matrix, obtaining the components with the greater variance. The PCA method goal is to extract attributes with strong correlation and substitute it by a set with less attributes. The Fig. 1 shows the load curve which describes the behavior of the variability percent with respect to the eigenvalues of the correlation matrix.

$$R = T\Lambda T' \tag{1}$$
$$\Lambda = diag(\lambda_1, \lambda_2, \lambda_2, \dots, \lambda_p) \tag{2}$$
$$S_o = diag(s_1, s_2, s_3, \dots, s_p) \tag{3}$$
$$s_i = \sqrt{var(X_i)} \tag{4}$$
$$Y = XS_o^{-1}T \tag{5}$$

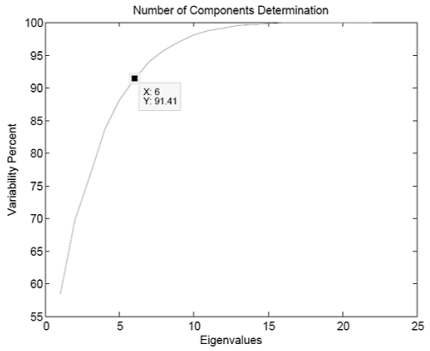

Fig. 1. Load curve to determine the number of components after applying PCA. With only six components there is a variability percentage of 91.41%.

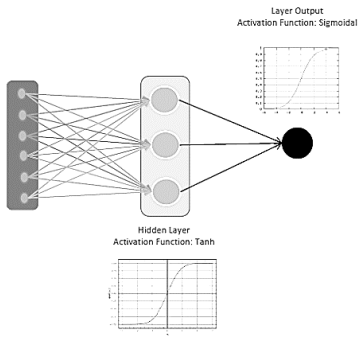

Fig. 2. MLP network implemented as solution to the problem of classification using PCA.

The equations (1)-(5), refers to the expressions used in PCA's application to the original dataset, where *X*, is the original dataset matrix; *R*, is the correlation matrix of *X*; *Λ*, is the diagonal *R* eigenvalues matrix; *T* are the *R* eigenvectors and *Y* is the new organized space.

The criterion used to determine the number of components that shape the new organized space is known as *Percentage Explained* [7], it uses the variability percentage depending on eigenvalues of the correlation matrix in the load curve.

The expression that gives origin to the variability percentage curve of the Fig. 1, is given by (6).

$$P_q = \frac{\sum_{i=1}^{q} \lambda_i}{\sum_{i=1}^{p} \lambda_i} * 100, \quad p > q \tag{6}$$

Using a criteria of 90%, as can be seen in Fig. 1, the 91.41% of the information is achieved if the first 6 components of the transformation are used.

The dataset is segmented in three subsets for training, validation and test. So, in 6 samples assigned to each patient; 3 are assigned to training set, for a total of 96 samples; 2 are assigned to validation set, for a total of 64 samples and 1 is assigned for test set, for a total of 32 samples.

IV. Classification using Multilayer Perceptron

The multilayer perceptron is basically a neural network that consists of an input neuron layer, a hidden neuron layer and an output layer. It's a learning method supervised by the error back-propagation algorithm. This solution approach adopts the convention, "1" for patients without Parkinson's and "0" for patients with Parkinson's.

A. Neural Network Design

Using nntraintool of Matlab®, there is a need to specify the number of neurons in the input, hidden and output layers, the size of the train, validation and test sets of the data, and the learning rate of the Network. Then the number of neurons is specified in the hidden layer based on the net size valid generalization given by Baum and Haussler [6], so for a projected 5 test errors, the maximum number of neurons in the hidden layer is M=13. However, several tests are performed with different sizes in order to determine the appropriate number of neurons in this layer in terms of the correct classification rate. Thus, based on experimental results the number of neurons in the hidden layer is determined in 3 for the training with PCA and 22 for the training that uses the complete dataset. The input layer is composed by 6 neurons for training applying PCA and by 22 neurons for the training that uses the complete dataset. In the output layer one neuron is used in both cases, given the fact that the network output is binary (healthy or with Parkinson's disease). In Fig. 2 is observed the structure of MLP network designed for training with PCA.

Other important parameter during the neural network design is the selection of the learning rate η. For this, is used the method suggested by Haykin [8], in which η must satisfy the condition given by (7).

$$0 < \eta < \frac{2}{\lambda_{max}} \tag{7}$$

Where λ_{max} is the maximum eigenvalue in the correlation matrix of the training input data set. In this case, λ_{max} is 1 given that all the eigenvalues of the main diagonal in the correlation matrix are 1; it´s a range of possible values for the learning rate: $0 < \eta < 2$.

The stopping criteria during the network training are selected as premature stop 10000 epochs, the mean square error of $1x10^{-4}$ and a value of gradient in error surface of $1\ x10^{-7}$.

To select a value of η, 20 experiments are performed with each value of η, varying this parameter from 0.1 to 2 with steps of 0.1. There is observed that performance in classification rate is better with values in the range $1 < \eta < 2$.

After analyzing the experiments results varying learning rate, the best results are observed with η values of 1.5, 1.6 and 1.7 with very similar between each other therefore taking learning rate $\eta = 1.7$ for the network training.

As can be observed in Fig. 2, in the output layer sigmoidal function is used as activation function, due to the fact that output only adopts two possible values: 0 or 1; which are precisely the minimal and maximum range values of sigmoidal function. In the hidden layer the activation function is the hyperbolic tangent, which in comparison to sigmoidal function has a better performance in the algorithm

TABLE I
MLP PARAMETERS

Parameters	
Inputs	6
Neurons in the output layer	1
Neurons in the hidden layer	3
Rate of Learning (η)	1.7
Performance	Classification rate
Stop criteria	
Epoch	10000
MSE	1e-4
Gradient	1e-7

convergence during the network training stage [8, pp. 200-206]. Table I, is a summary of the parameters selected in the construction of the MLP network implemented.

B. Training with complete dataset

As mentioned above, the best results are obtained with the learning rate of 1.7. Therefore training with different number of neurons in the hidden layer is performed, following the Baum and Haussler generalization, mentioned in paragraph A.

Fifty experiments are performed for each hidden layer size (3, 8, 13 and 22 neurons), and both average of mistakes and correct classification rate are obtained.

Given that the classification results thrown by the MLP are not exactly '1' or '0', a condition is implemented in order to determine when a value between 0 and 1 can be a healthy or a Parkinson's patient. This condition is as follows: If the value is greater than 0.6 is assumed like 1, and if the value is less than 0.4 is assumed like 0. In addition to set an errors count pattern in the validation stage the following assessment is carried out: There is a classification error if validation data is 1 and the network output is less than 0.6 or if validation data is 0 and the network output is greater than 0.4. Based on these considerations, the results for different number of neurons in the hidden layer are shown in Table II, where NHL: Neurons in hidden layer; AM: Average of Mistakes; CACR: Correct Average Classification Rate.

As can be seen in Table II, the correct average classification rate increases as the number of neurons in the hidden layer increases, and the best result occurs with 22 neurons in this layer with an average CACR of 90,24%. In Fig. 3 is shown one of the best results obtained with 22 neurons in the hidden layer, it has just 2 classification errors and a correct classification rate of 96,875%.

TABLE II
MLP RESULTS WITH COMPLETE DATASET

Parameter				
NHL	**3**	**8**	**13**	**22**
AM	8,9	7,04	6,44	6,24
CACR (%)	86,09	89	89,94	90,24

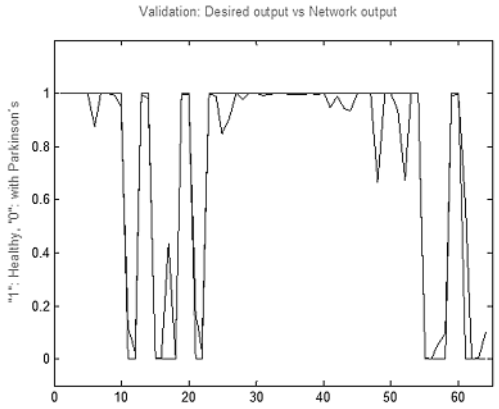

Fig. 3. One of the best results with 22 neurons in the hidden layer. It has a correct classification error rate of 96,875 % (just two errors).

TABLE III
MLP RESULTS WITH PCA USING DIFFERENT OPTIMIZATION METHODS

Parameter				
Opt. Method	**GD**	**SCG**	**RP**	**LM**
AM	6,53	8,14	8,26	8,84
CACR (%)	89,79	87,28	87,09	86,19

C. Training applying PCA

The training applying PCA is divided in three stages: The first stage as mentioned above is to determine the best learning rate with which the synaptic weights will be modified. After analyzing the experimentation results, the selected learning rate for network training is η = 1.7. The second stage is use different optimization methods to find a local minimum. These methods are:

- Gradient Descent (GD)
- Scaled Conjugated Gradient (SCG)
- Levenberg-Marquardt (LM)
- Resilient (RP)

Fifty experiments are performed for each optimization method with η = 1.7 and 3 neurons in the hidden layer obtaining the results that are shown in the Table III.

As can be seen, the best results are obtained with Gradient Descent as optimization method. Then, the third stage is use GD as optimization method and to experiment with different number of neurons in the hidden layer based on Baum and Haussler net size generalization, which thrown M=13 as the maximum number of neurons in the hidden layer. Therefore, fifty experiments are realized with 8 and 13 neurons in the hidden layer (NHL) in order to compare the results with the 3 initial hidden neurons. The results of it are shown in Table IV.

The obtained results shows that, increase the number of neurons in the hidden layer doesn´t have a satisfactory performance in terms of classification rate.

TABLE IV
MLP RESULTS WITH PCA, GD AND OTHER NUMBER OF NEURONS IN THE HIDDEN LAYER

	Parameter	
NHL	**8**	**13**
AM	8,52	8,94
CACR (%)	86,69	86,03

V. CLASSIFICATION USING SELF-ORGANIZING MAP

The self-organizing maps, is one of the unsupervised learning methods most used in neural networks, as the name implies does not have an external supervisor to verify and monitor the learning process, therefore the aim of the neural network involves the search for correlations, characteristics or behavior patterns in the input data; this indicates that unsupervised learning generates satisfactory results when regularities are present in the input data.

Kohonen's maps or self-organizing maps, used as a criterion for modifying connections, or synaptic weights, use the approach proposed by Donald Hebb in 1949, known as the Hebb Rule: "Let us assume that the persistence or repetition of a reverberatory activity (or "trace") tends to induce lasting cellular changes that add to its stability.… When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased.", In other words, the modification of the synaptic weights of the neuron depends of the received stimulation set of the neurons around it.

$$\Delta W_{ij} = a_i \cdot a_j \tag{8}$$

Equation (8) shows the variation of synaptic weight between neurons *i* and *j*, where a_i and a_j are activation values of neuron i and j respectively given great importance to the topology of the neural network, because the way in which network are interconnected, determine the behavior of the synaptic weights and thereby the performance of the network.

However, to determine the topology of the network, it must be related to the lateral interaction model [9] that explains how to perform the interaction between neurons on the self-organizing map. This model proposes two layers: the input layer and competitive layer. Each element of the input layer is interconnected to each neuron in the competition layer. Each neuron of the network is connected to a competition set of neighboring neurons (topology) and is able to excite a set of neighboring neurons (neighborhood function) in order to generate a learning method called winner takes all, wherein each neuron modifies its own to achieve its synaptic weights on the neurons excitation closest, but also seeks an inhibitory effect in their neighbors that are far away.

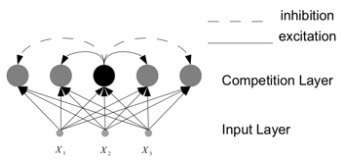

Fig. 4. Self-Organizing Map.

4. Toolbox available under GNU General Public License at: http://www.cis.hut.fi/projects/somtoolbox/
5. Fig. 4-6. Taken from somtoolbox manual, Available: http://www.cis.hut.fi/projects/somtoolbox/download/.

This effect is achieved using a neighborhood function (Fig. 4).

A. Neural Network Design

The proposed solution is implemented using somtoolbox[1] [10] tool developed by the University of Helsinki. This tool requires different parameters for neural network construction, including define the function that determines the competitive process, the neighborhood function that acts in the cooperative process, the learning rate function η, which acts in the adaptive process, the map size, the initialization method of the synaptic weights and the training method (presentation of data during network training).

The addressing of this solution is using PCA as indicated in section II, using the same sets of training, validation and testing. To determine the map size the following expression is used (9) [10]:

$$size = 5\sqrt{n} \tag{9}$$

Where n = 96, which is the number of instances used in network training that gives an estimated size for the map of 49 neurons.

With this network size the relationship between the sides of the map is established as the root square of the ratio between the two largest eigenvalues (in this case the two highest eigenvalues are 12.83 and 2.49) of the whole correlation matrix input data [10]. According to the above, the relationship that should exist between the two sides is 2.27. Given the size of the map values and the relationship between its sides, the size assigned to the map is 10 x 4 neurons.

The function used by the toolbox at the competition stage is a measure of the distance through the modified Euclidean norm (10):

$$\|x - m\|^2 = \sum_{k \in K} w_k (x_k - m_k)^2 \tag{10}$$

In the competition layer, somtoolbox provides two interaction topologies in the neighborhood of neurons, for the proposed solution, a hexagonal structure is used; as can be seen in Fig. 5. Some experiments are performed using Gaussian and Cutgauss neighborhood functions, available in the toolbox[2] [11], finally Gaussian function is select as a neighborhood function (Fig. 6). and the Mexican hat (Ricker wavelet) is included in the toolbox to compare results.

Fig. 7 shows the learning rate η functions used by the toolbox. The design of the proposed network function is selected *inv* because it converges in less time to $\eta \approx 0$. The training is realized in two steps: a coarse adjustment step in which the learning rate function has a large variation and finishes with a fine adjustments step in the values of η.

According to information provided by the toolbox developers, using batch training method reduces convergence times of the network [10] [11]; likewise, expressions are used to estimate the number of epochs that are carried out for the network learning and the initial radius of the neighborhood function [9] [11], which in this case is 23 epochs and the initial neighborhood radius of 2.5. The parameters used in the design of self-organizing map are summarized in Table V.

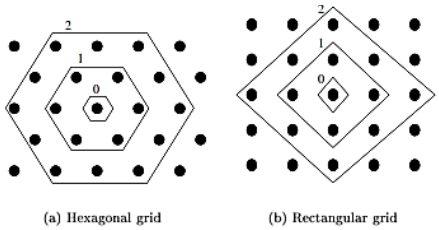

Fig. 5. Interaction topologies in the neighborhood.

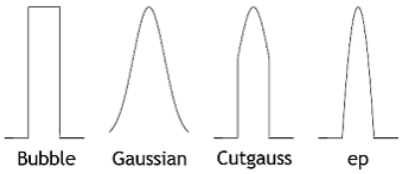

Fig. 6. Neighborhood Functions.

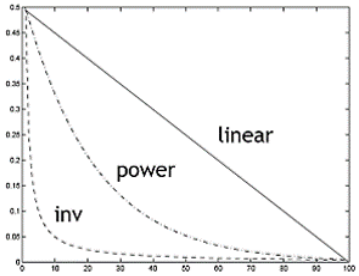

Fig. 7. Learning rate functions.

TABLE V
SOM PARAMETERS

Parameters	
Inputs	6
Map Size	40 (10 x 4)
Competition Function	Modified Euclidean Distance
Neighborhood Function	Gaussian
Neighborhood Topology	Hexagonal
Function η	Inverse
Initial Neighborhood Radio	2.5
Final Neighborhood Radio	1
Performance	Classification rate

B. Training with the complete dataset

This training is made with the following parameters: Gaussian neighborhood function, hexagonal cells neighborhood topology and 10x4 Map Size. So, the obtained result is an average of mistakes of 14.86 for a correct average classification rate of 76.78%.

C. Training applying PCA

The training of the self-organized map using only the 6 attributes after the Principal Components Analysis is performed for three cases:

- Changing the map dimensions
- Changing the neighborhood function
- Changing the cells neighborhood topology

1) Changing the map dimensions

In this experimentation, the initial map size of 10x4 is changed to 10x5 in order to compare this results with the original dimension which is sized based on the two higher eigenvalues as explained previously. The results with 10x5 map size is an average of mistakes of 28.98 for a correct average classification rate of just 54.72%. It is the worst result obtained, which suggests that the adequate relationship between the sides of the network is the exposed by de Faria *et al* [10].

2) Changing the neighborhood function

There is two neighborhood functions that are used: Gaussian, and the Mexican hat (Ricker wavelet). This function is described by (11).

$$mh(Ud,r) = -2e^{-(Ud^2+r^2)} - 0.5e^{-(Ud^2+r^2)/3} \quad (11)$$

Where, Ud is the topological distance and r is the neighborhood radius, both used by somtoolbox.

The Mexican hat function is introduced in the toolbox because his inhibitory behavior to the near cells is stronger than the Gaussian function, given that using the Mexican hat function the lateral interaction has the following behavior: If a cell is closed to the analyzed cell, it receives a strong influence of it. As the distance between the cells increases, decrease the influence until become even negative [12], while the Gaussian function takes always positive values in both cases. Can be said that the Mexican hat function has a stronger inhibitory behavior than Gaussian function. Mexican hat function can be appreciated in Fig. 8.

For both cases an inverse learning rate is used; the initial and final neighborhood radio are 2.5 and 1 respectively as mentioned above in section A. Thus, the obtained results for fifty experiments are summarized in Table VI.

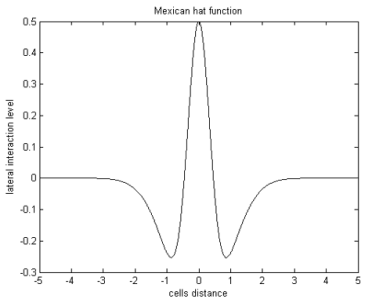

Fig. 8. Mexican hat function. It can be seen how cells closed to the analyzed cell (distance zero) receives a strong influence of it, while more distanced cells experiments inhibition (negative Y axis values), and finally too far cells don´t experiment any influence of the analyzed cell (cero Y axis values).

TABLE VI
SOM WITH PCA CHANGING THE NEIGHBORHOOD FUNCTION

Parameter		
Neigh. F.	**Gaussian**	**Mex. Hat**
AM	12,94	13,28
CACR (%)	79,78	79,25

3) Changing the cells neighborhood topology

In this case the rectangular topology is used as can be seen in Fig. 5, with a 10x4 size map, using two neighborhood functions, gauss and cutgauss (see Fig. 6). For every neighborhood function 25 experiments are performed, the obtained results are summarized in Table VII.

TABLE VII
SOM WITH PCA AND RECTANGULAR TOPOLOGY CHANGING THE NEIGHBORHOOD FUNCTION

Parameter		
Neigh. F.	**Gaussian**	**Cutgauss**
AM	12	12,52
CACR (%)	81,25	80,43

VI. COMPARISON BETWEEN MLP AND SOM OBTAINED RESULTS

In Tables VIII and IX are summarized all the obtained results for MLP and SOM with and without applying PCA. As can be seen the best result is MLP without PCA, 22 neurons in hidden layer and Gradient Descent as optimization method.

TABLE VIII

MLP Results				
Opt. Method	**NHL**	**PCA**	**AM**	**CACR(%)**
GD	3	No	8,9	86.09
GD	8	No	7.04	89
GD	13	No	6.44	89.94
GD	22	No	6.24	90.24
GD	3	Yes	6.53	89.79
SCG	3	Yes	8.14	87.28
RP	3	Yes	8.26	87.09
LM	3	Yes	8.84	86.19
GD	8	Yes	8.52	86.69
GD	13	Yes	86.69	86.03

TABLE IX

SOM Results					
Net. Size	**Neigh. Function.**	**Topology**	**PCA**	**AM**	**CACR(%)**
10x4	Gaussian	Hex.	No	14.86	76.78
10x5	Gaussian	Hex.	Yes	28.98	54.72
10x4	Gaussian	Hex.	Yes	12.94	79.78
10x4	Mex. hat	Hex	Yes	13.28	79.25
10x4	Gaussian	Hex.	Yes	12	81.25
10x4	Cutgauss	Hex.	Yes	12.52	80.43

VII. CONCLUSION

Applying a Multilayer Perceptron with the complete dataset (without preprocessing of data), 22 neurons in hidden layer and Gradient Descent as optimization method, the neural network has the best performance with a highly and significant correct average classification rate of 90.24% which is slightly less to the result obtained by Little *et al* [2] but using a less expensive computational algorithm. On the other hand, the best classification result reached is a correct classification rate of 96,875%; therefore MLP without PCA can be taken into account for Parkinson's diagnosis. For this dataset, best diagnosis results are obtained using MLP than using SOM. The inclusion (in somtoolbox) of the Mexican hat, from which it was the hypothesis of has better results than the Gaussian function by his inhibition shape, does not really improvements in the classification performance. Apply data preprocessing trough PCA doesn't have a significant improve in the performance of the neural network, for both MLP and SOM.

REFERENCES

[1] Little MA, McSharry PE, Roberts SJ, Costello DAE, Moroz IM. 'Exploiting Nonlinear Recurrence and Fractal Scaling Properties for Voice Disorder Detection'. BioMedical Engineering OnLine 2007, 6:23 (26 June 2007)

[2] Little, M.A.; McSharry, P.E.; Hunter, E.J.; Spielman, J.; Ramig, L.O., "Suitability of Dysphonia Measurements for Telemonitoring of Parkinson's Disease," *Biomedical Engineering, IEEE Transactions on* , vol.56, no.4, pp.1015,1022, April 2009

[3] Fritsch, T.; Kraus, P. H.; Przuntek, H.; Tran-Gia, P., "Classification of Parkinson rating-scale-data using a selforganising neural net," *Neural Networks, 1993., IEEE International Conference on* , vol., no., pp.93,98 vol.1, 1993

[4] Voros, T.; Keresztenyi, Z.; Fazekas, Cs; Laczko, J., "Computer Aided Interactive Remote Diagnosis Using Self-Organizing Maps," *Engineering in Medicine and Biology Society, 2004. IEMBS '04. 26th Annual International Conference of the IEEE* , vol.2, no., pp.3190,3193, 1-5 Sept. 2004

[5] Geman, O., "A fuzzy expert systems design for diagnosis of Parkinson's disease," *E-Health and Bioengineering Conference (EHB), 2011* , vol., no., pp.1,4, 24-26 Nov. 2011

[6] Baum, E.; Haussler, D., "What Size Net Gives Valid Generalization?" Issue of Neural Computation, MIT Press, January 1989.

[7] Grané, A., "Análisis de Componentes Principales", Departamento de Estadistica, Universidad Carlos III de Madrid. [Online]. Available: http://halweb.uc3m.es/esp/Personal/personas/agrane/ficheros_docencia/MULTIVARIANT/slides_comp_reducido.pdf.

[8] Haykin, S. "Neural Network: A comprehensive foundation", 2nd ed. Singapore: Pearson Education, 2005. pp. 150-157.

[9] Isasi, P. and Galván, I. "Redes de Neuronas Artíficiales, un enfoque práctico", Madrid:Pearson Education, 2004. pp. 123-144.

[10] De Faria, E., Portes, M., Gonzalez, J., Portes Marcio P., Pinto J, "Introduçao ao Toolbox de Redes Neurais de Kohonen". Centro Brasileiro de Pesquisas Físicas. Universidade Federal do Espírito Santo., Rio de Janeiro, Brasil, 2010.

[11] Vesanto, J., Himberg J., Alhoniemi, E., Parhankangas, J. "Som Toolbox for Matlab 5". Som Toolbox Team. Helsinki University of Tecnology, April, 2000.
Available:http://www.cis.hut.fi/projects/somtoolbox/download/.

[12] Isasi, P. and Galván, I. "Redes de Neuronas Artíficiales, un enfoque práctico", Madrid:Pearson Education, 2004. pp. 130-131.