ORB ACADEMIC PUBLISHER

Int. J. Comput. Sci. Theor. App., 2015, vol. 3, no. 3., p. 74-86.
Available online at www.orb-academic.org

International Journal of
Computer Science:
Theory and Application
ISSN: 2336-0984

Using Neural Networks and Feature Selection Algorithm to Perform Early Diagnosis of Alzheimer and Mild Cognitive Impairment Disease

Lara Dantas*, Mêuser Valença

Polytechnic School, University of Pernambuco, Recife, Brasil
Email: ldc@ecomp.poli.br, meuser@ecomp.poli.br
**Corresponding author.*

Abstract

Alzheimer's disease (AD) is now considered the most common type of dementia in the population. Although it is a degenerative and irreversible disease, if diagnosed early, medications may be administered to slow the progression of symptoms and provide a better quality of life for the patient. Herbert et al. and Gòmez conducted studies with classifiers contained in the software Weka using a database with values of 120 blood proteins, and they noticed that they could classify the patient may or may not be diagnosed with AD with an accuracy rate of 93% and 65%, respectively. Thus, this study aims to use neural networks such as Multi-layer Perceptron, Extreme-learning Machine and Reservoir Computing to perform early diagnosis of a patient with or without AD and Mild Cognitive Impairment (MCI), another common type of disease. This article also envisions to utilize the Random Forest Algorithm to create a new signature with 3 protein. Through experiments it can be concluded that the best performance was obtained with the MLP and the signature with 3 proteins doesn't have a great accuracy when compared to those available in the literature.

Keywords

Neural Networks — Alzheirmer's Disease — Mild Cognitive Impairment Disease — Feature Selection Algorithms — Random Forest Algorithm.

1. Introduction

Most developed countries are undergoing a major demographic shift. The oldest segments of the population are growing at a faster rate, and therefore, there is a constant increase in age-related diseases, especially progressive dementia disorders. First described by psychiatrist Alois Alzheimer in 1907, Alzheimer's Disease (AD) is, today, the most common cause of dementia in the elder population [1].

According to the Brazilian Institute of Geography and Statistics (IBGE) and the World Health Organization (WHO), there is 1.2 million people with AD in Brazil. It is believed that only 5% of the patients developed the disease at an early stage, i.e., before 65 years of age. In patients where the AD started after 65 years old, it is estimated that between 10% and 30% of these cases started after 85 years old [2].

AD is a degenerative disease that causes irreversible death of several brain cells and the neurons. The patient suffering from this disease has a brain with microscopic pathologic lesions, known as neuritic plaques, and neurofibrillary tangles [3]. In addition, the brain of a person with Alzheimer's is much smaller than the brain of a healthy person, as it is shown in Fig. 1.

This disease develops in each patient in a unique way; however, there are several symptoms common to all of them, e.g., loss of memory, language disorders, depression, aggression, among others. Initially, the patient loses episodic memory, i.e., memory that holds information of events and their spatio-temporal relations. Thus, the old facts and the facts that just happened are easily forgotten. With the progress of the disease, semantic memory is also lost, i.e., lexical knowledge, rules, symbols are forgotten and the patient begins to lose its cultural identity [4].

The diagnosis of AD is often performed late since this disease can be confused with several other types of dementia and even the normal symptoms of aging. Although it is an irreversible disease, if it is discovered in its early stage, medications may be administered to slow the progression of symptoms and prolong the patient's welfare [5]. Thus, it is extremely important

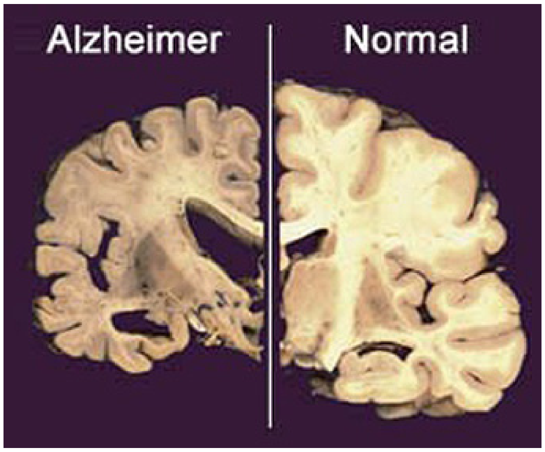

Figure 1. *Brains illustrations. In the left side the one of a person with AD and in the right side a brain of a healthy person.*

that mechanisms are developed for an earlier prediction of AD in the whole population.

Another common type of disease is the Mild Cognitive Impairment (MCI). The MCI causes cognitive changes that are noticed by the individuals experiencing them or to other people that live with the patient. However, this changes are not severe enough to interfere with daily life or independent function. People who are diagnosed with MCI have an increased risk of eventually developing AD.

In the literature, Ray et al. [6] conducted a study using a database of 120 samples of proteins contained in plasma of several patients. They concluded in the research that a combination of 18 out of the 120 available proteins enabled the realization of early diagnosis of AD with an classification rate of 91% using a set of tests with data from 92 patients who were diagnosed with or without AD.

In addition, he also used another set of tests containing data from 47 patients diagnosed with Mild Cognitive Impairment (MCI). For this set, the classification rate was 81%. These values were calculated from the average success rates found for all classifiers used in clinical trials for both sets [7].

Afterwards, Gòmez et al. conducted a study using 20 different classifiers available in the software Weka and defined various signatures with 18, 10, 6 and 5 proteins. These proteins were all contained in the set described by Ray et al. The 10 proteins signature reached a classification rate of 89% using the AD test set and 66% for the MCI test set. Furthermore, the 5 proteins signature reached a classification rate of 93% using the AD test set and 65% for the MCI test set. These success rates were also calculated from the average values of the 20 classifiers used [7].

Dantas and Valença [1] made a significant contribution. In the work mentioned, it was used the Random Forest Algorithm to create a new signature with 10 proteins and test its accuracy with 2 topologies of neural networks: Multi-Layer Perceptron and Reservoir Computing. After all the experiments, it was statistically proven that the new signature has a higher classification rate for the diagnosis of AD and MCI.

Another work of Dantas and Valença [8] used the same Random Forest Algorithm and created a new signature with 5 proteins. Besides that, the accuracy of this new signature and the one proposed in [1] and in [7] was tested with a new topology of Neural Network: The Extreme-Learning Machine [9]. The results were proved statistically and the signature with 5 proteins has also a higher classification rate for both diagnosis: AD and MCI.

Recently, Dantas and Valença [10] tried to use another feature selection algorithm to create two new signatures with 10 and 5 proteins and test the results with the 3 topologies of neural networks: Multi-layer Perceptron [11], Reservoir Computing [12] and Extreme-Learning Machine [9]. But, the statistic tests proved that the signatures created with the Random Forest Algorithm in [8] were better than the new ones.

This paper aims to use the Random Forest Algorithm to create new signatures with 3 proteins to perform early diagnosis for AD and MCI. Beyond that, the accuracy of each signature will be tested using the Multi-layer Perceptron [11], Reservoir Computing [12] and Extreme-Learning Machine [9].

It is very important to try to minimize the number of proteins in the signature, since the higher this number, the greater the cost of the exam to extract these proteins.

This article is organized into several sections. The first contains information about the neural network topologies that will be used. The next section describes the methodology used throughout this work, i.e., what database is used and how it is organized, the experiments and statistical analysis that were performed. Finally, there is a section that displays the results and the last one contains the conclusions obtained in this work.

2. Artificial Neural Networks

The Artificial Neural Networks (ANN) are models that intend to simulate the behavior of the human brain. The ANN contains simple processing units that are interconnected in order to process information. The knowledge of this model is stored in the weight contained in each of the connections between the artificial neurons.

The ANNs have been extensively used in many fields due to their ability to approximate complex nonlinear functions. These models have some advantages such as generalization, adaptability, ability to learn from examples.

Over the years, many studies have emerged in this field and several ANN topologies were created. In this paper, some of them will be discussed in the next sessions.

2.1 Multi-Layer Perceptron

One of neural networks topologies used in this paper was the Muti-Layer Perceptron (MLP). This network has the advantage of having intermediate layers. Thus, this characteristic guarantees that this neural network can approximate any continuous function as long as it has at least one intermediate layer, or

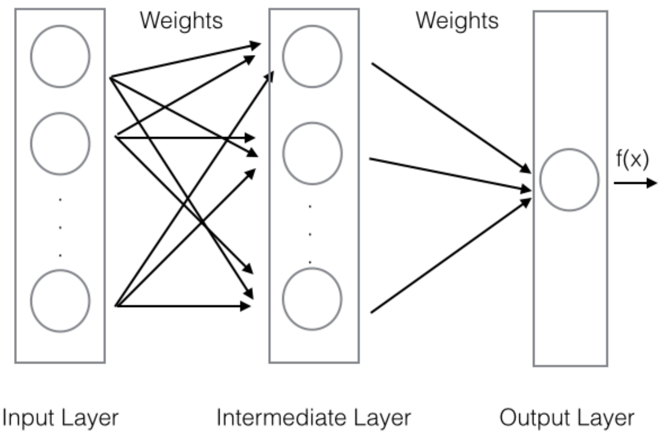

Figure 2. *MLP with three layers*

any mathematic function if the number of intermediate layers is more than one.

All the connections between the neurons of the MLP have a weight that, initially, has a random value but that during the training will be optimized. These neurons are disposed in three kinds of layers that is listed below:

- **Input layer:** Represents the input variables of the problem;
- **Intermediate or hidden layer:** This layer is responsible for the capacity of the MLP in solving non linear problems;
- **Output layer:** Represents the output variables of the problem

Fig. 2 shows an example of this network. As well as other topologies of neural networks, the MLP also needs a training algorithm to optimize the weights of the MLP. In this paper, it was used the Back-propagation, a gradient-based algorithm [13].

2.1.1 Back-propagation algorithm

This algorithm is divided in two phases. In the first one, the forward phase, it happens the progressive signal propagation, i.e., it goes from the input layer to the output layer. In this phase, the weights are not changed and the output of the neural network and the error are calculated.

At this moment, the second phase, backward, of the algorithm is initiated. During this stage, the back-propagation error occurs, that is, based on the calculated output error, the weights connecting the hidden layer to the output layer are adjusted. This is accomplished using Equation (1) [13].

$$W_{i,j}^{m}(t+1) = W_{i,j}^{m}(t) + \alpha\delta_i^m f^{m-1}(net_j^{m-1}) + \beta\Delta W_{i,j}^{m}(t-1) \quad (1)$$

where

- α: represents the learning rate of the algorithm;

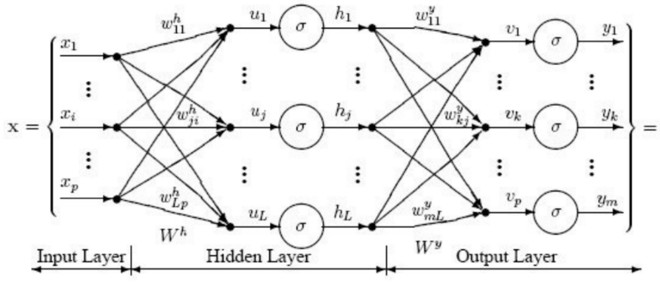

Figure 3. *Single Layer Feedforward Network*

- β: represents the momentum rate;
- δ: represents the sensibility that can be calculated following Equation (2) to the neurons in the output layer and using Equation (3) to the neurons in the hidden layers [11].

$$\delta_i^m = (d_i - y_i)f'(net_i) \quad (2)$$

$$\delta_j^{m-1} = f'^{m-1}(net_j^{m-1})\sum_{i=l}^{N} w_{ij}^m, \delta_i^m \quad (3)$$

More detailed information about the execution of this algorithm can be found in the literature [13][14][11].

2.2 Extreme-learning Machine

With the increase in the use of neural networks in many applications, it was noted that the learning speed of feedforward networks, such as MLP, usually was very slow. This problem also makes it difficult to use these neural networks for problems that require real-time response .

The main reason for this speed problem was that almost all ANNs were trained using gradient based algorithms. These kind of algorithms have several issues, such as the slower speed in the learning process, overfitting and local minimums.

In order to try to solve this problem, Huang et. al. created in 2004 the Extreme Learning Machine (ELM), a new training algorithm for Single-hidden Layer Feedforward Neural Networks (SLFNs) [9]. This algorithm randomly chooses the input weights and analytically determines the output weights of a SLFN [15]. The architecture of a SLFN is shown on Fig. 3.

ELM can be modeled following the Equations (4) and (5) [9]:

$$\sum_{i=l}^{\tilde{N}} \beta_i g_i(x_j) = \sum_{i=1}^{\tilde{N}} \beta_i g(w_i.x_j + b_i) = o_j, j = 1,...,N \quad (4)$$

$$\sum_{i=l}^{\tilde{N}} \beta_i g(w_i.x_j + b_i) = t_j, j = 1,...,N \quad (5)$$

where

- (x_j, t_j): N input patterns;
- w_i: Weight vector of the neuron i from the hidden layer;
- b_i: bias of the neuron i from the hidden layer;
- $\tilde{N}$: number of neurons in the hidden layer;
- β_i: Weight vector between the hidden neuron i and the output layer.

The ELM also can be demonstrated in a matrix form as described in Equation (6) [15].

$$H\beta = T \tag{6}$$

where

- H is expressed in Equation (7) [9];
- β is expressed in Equation (8) [9];
- T is expressed in Equation (9) [9];

$$H(w_1,...,w_{\tilde{N}},b_1,...,b_{\tilde{N}},x_1,...,x_{\tilde{N}}) = \begin{bmatrix} g(w_1.x_1+b_1) & \cdots & g(w_{\tilde{N}}.x_1+b_{\tilde{N}}) \\ \vdots & \cdots & \vdots \\ g(w_1.x_N+b_1) & \cdots & g(w_{\tilde{N}}.x_N+b_{\tilde{N}}) \end{bmatrix} \tag{7}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_{\tilde{N}}^T \end{bmatrix}_{\tilde{N}xm} \tag{8}$$

$$T = \begin{bmatrix} t_1^T \\ \vdots \\ t_{\tilde{N}}^T \end{bmatrix}_{Nxm} \tag{9}$$

The ELM pseudo code training algorithm is available in Algorithm 1.

Algorithm 1: ELM Pseudo code

```
Randomly select values for the weights $W_i$ and bias $b_i$, i = 1, ... , N;
Calculate the output matrix H of the hidden layer;
Calculate the output weights $\beta = H^{\dagger}T$;
```

where $H^{\dagger}$ is the Moore-Penrose Pseudo Inverse [9] and needs to satisfy the following properties:

- $HH^{\dagger}H = H$;
- $H^{\dagger}HH^{\dagger} = H^{\dagger}$;
- $(HH^{\dagger})^T = HH^{\dagger}$;
- $(H^{\dagger}H)^T = H^{\dagger}H$.

This pseudo inverse can be efficiently calculated using the Single Value Decomposition (SVD) [15].

Furthermore, the ELM offers significant advantages such as ease of use, better generalization performance with much faster learning speed and it can be used in real-time applications.

A remarkable point to be considered is that the gradient based algorithms can be used for training neural networks with multiple hidden layers. In contrast, the ELM can only be used in SLFNs. However, as previously mentioned, with only a single hidden layer, the ANN can approximate any continuous function.

Thus, the use of this algorithm in this paper is valid, since it is a problem that can be solved with SLFNs networks.

2.3 Reservoir Computing

Recurrent Neural Networks (RNN) were created to enable the solution of dynamic problems. This is accomplished through a feedback of a neuron in a layer i to that found in some previous layer, $i - j$. This neural network topology has a better resemblance to the operation and behavior of the human brain [16].

In 2001, a new approach for the design of the training of a RNN was proposed by Wolfgang Mass called Liquid State Machine (LSM) [17]. At the same time, but independently, the same approach was described by Herbert Jaeger and called Echo State Machine (ESN) [18].

Both ESN and LSM networks have the Echo State Property (ESP) [19], i.e., due to the recurrent network connections, information from previous entries are stored. However, these data are not stored for an infinite period of time, and as well as the human brain, old information must be forgotten over time. Thus, the neural network has a rich set of information from the past and present therefore enhancing its applicability to dynamic systems [20].

In 2007, Verstraeten coined the term Reservoir Computing (RC) that unified the concepts described in ESN and LSM. Since then, this term is used in literature to illustrate learning systems which are represented by a dynamic recurrent neural network [12].

The RC is composed of three parts: an input layer, which as the MLP, represents the input variables of the problem, a reservoir, which can be seen as a large distributed and dynamic RNN with fixed weights, and a linear output layer called readout.

Fig. 4 represents the RC topology with two neurons in the input layer, three in the reservoir and one neuron in the output layer.

Although the problem addressed in this work is not dynamic, it was decided to test this neural network topology in order to verify the processing time and the memory effect of the RC and if it would have some positive impact on the diagnosis of AD.

The RC used in this work was based on the ESN approach and was developed in the Java programming language to make use of the object-oriented paradigm. This framework was created in order to solve classification and prediction problems and

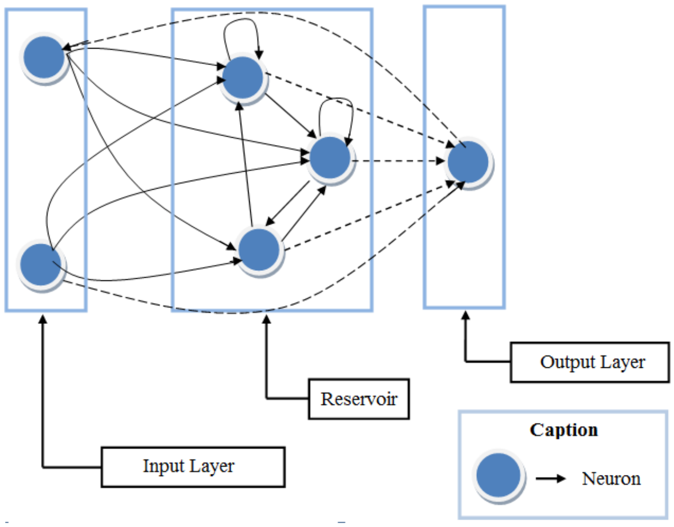

Figure 4. *RC architecture. The dashed lines represent the weights that should be adjusted during the training of the network.*

it was validated by three Benchmarks: Iris species, Thyreoide, Cancer and Diabetes.

The training algorithm for the RC was the same as the ELM, i.e., the outputs are calculated using the Moore-Penrose generalized inverse and it is more detailed in the next section.

2.3.1 Construction and Simulation of RC

The first step in order to prepare the RC and perform the data set classification is configure its architecture. Thus, it is necessary to define the number of neurons that will be used in the input, output and in the reservoir layers.

Furthermore, several RC parameters should also be determined. Being a recent methodology, there are no studies that prove how many neurons in the reservoir are necessary so that the neural network has better performance, or the rate of connectivity between these neurons. Therefore, for this work, values were defined based on some empirical tests performed.

Once the architecture of the neural network is determined, the next step is to generate the weight matrices connecting the input layer to the reservoir, W_{in}, and the matrix with the weights between neurons in the reservoir, W_{res}. Both matrices are generated with random values between -1 and 1.

Studies claim that the matrix W_{res} must have a spectral radius equal to 1 to provide a more numerical stability [21], i.e., when W_{res} is initialized, it must have its values changed as follows:

- Initially it must be decomposed into singular values;
- Then, W_{res} should have its values changed until the maximum value of the main diagonal of the eigenvalues matrix is less than or equal to 1.

To perform the simulation of the RC, the database is divided into three sets: training, used to perform the update of the states of the neurons of the reservoir, cross-validation, used to stop the training of the neural network, and test set, used to calculate the RC classification rate [22].

The states of the neurons in the reservoir must be initialized to zero. Since this is a recurrent network and RC stores its states (M_{est}) in a matrix, it is necessary that the final values found by the network are not so influenced by this initialization. Therefore, the literature suggests that before start training, a set of cycles called warm up is executed in order to perform updates in the states of the neurons in the reservoir and overlook the influence of the initial value [12]. The states are updated according to (10) [12]:

$$x[k+1] = f(W_{res}x[k] + W_{in}u[k]) \tag{10}$$

where $W_{in}u[k]$ represents the matrix containing the result of the product of the values derived from the input layer by the weights connecting these neurons to the reservoir at a time k and $W_{res}x[k]$ is the matrix with the states of the neurons from the reservoir at the same time k. The result will be assigned to $x[k+1]$, i.e., the state of the neuron RC in an instant forward will be the result of calculating the activation function of the neuron from the sum of the two parcels described above. In this work, the activation function used was the hyperbolic tangent according to equation 11[11].

$$f(net_i) = \frac{e^{net_i} - e^{-net_i}}{e^{net_i} + e^{-net_i}} \tag{11}$$

where, y is the output value and net_i is the weighted average of the weights with the entries of the ith neuron.

Once the period of warm up is over, the training of the RC can be initialized. The first step should be to load the training set and perform the update of the states of the reservoir, noting that the matrices W_{in} and W_{res} should not be changed. They are randomly generated during construction of the RC, as described in the previous section, and should not be adjusted.

Still during training, the weights matrix that connects the neurons of the input layer to the output (W_{inout}) and the one that connects the reservoir to the output must be calculated by the pseudo-inverse of Moore-Penrose. As they are non-square matrices and their determinants can approach zero, it is necessary to calculate the pseudo-inverse.

At the end of each training cycle, a cross validation cycle should be initiated. This process should be repeated until the stopping criteria is reached and the training set is finalized. During the process of cross-validation, the matrices W_{inout} and W_{out} should remain being readjusted.

When the process of training is finished, the testing process begins. The set of tests is presented to the RC and at this time, all the weights matrices, W_{in}, W_{res}, W_{inout} and W_{out}, should remain unchanged, as the matrix M_{est}. At this point, the classification error is calculated. These values will be used in the future to make the necessary comparisons.

The behavior of the RC can be best viewed through Algorithm 2[12].

Algorithm 2: Pseudocode of RC

```
Set the number of neurons in the input layer ;
Set the number of neurons in the reservoir layer ;
Set the number of neurons in the output layer ;
Randomly generate the weights of $W_{in}$ between -1 e 1;
Randomly generate the weights of $W_{res}$ between -1 e 1;
Normalize the weights of $W_{res}$ so that the spectral radius
  of the matrix is smaller than or equal to 1;
while until the end of the number of warm up cycles do
    updates the states of the neurons of the RC;
end
while until the stopping criterion is reached do
    for each value of the input set do
        updates the states of the neurons of the RC;
    end
    Calculates the Moore-Penrose matrix inverse to find
      the weights connecting the RC to the output layer;
    Calculates the Moore-Penrose matrix inverse to find
      the weights connecting the input layer to the output
      layer;
    for each value of the cross-validation set do
        updates the states of the neurons of the RC;
    end
    Calculates the output values of the RC;
    Calculates the RMSE;
    Checks if the stopping criterion has been reached;
end
for each value in the set of tests do
    updates the states of the neurons of the RC;
end
Calculates the output values of the RC;
Calculates the accurancy rate;
```

3. Random Forest Algorithm

Although the Random Forest (RF) Algorithm is an excellent classifier, this technique can be used to rank the importance of variables in a classification problem [23].

To measure the importance of each variable in a data set D_n, it has to fit a random forest to the data. The data set can be expressed as shown in Equation (12) [24].

$$D_n = (X_i, Y_i)_{i=1}^{n} \qquad (12)$$

where

- X is the training set;
- Y is the responses set;
- n is the number of the examples in the data set.

Algorithm 3 shows the pseudo code used to calculate the Variable Importance (VI) with the RF. It is important to mention that this algorithm uses the Out-of-bag (OOB) error estimation in the formula to measure the VI [23].

Algorithm 3: Pseudocode of the calculus of VI using RF

```
for each tree t do
    Consider the associated $OBB_t$ sample;
    Denote by $errOBB_t$ the error of a single tree t on this
      $OBB_t$ sample;
    Randomly permute the values of $X^j$ in $OBB_t$ to get a
      perturbed sample denoted by $OBB_t^j$, the error of
      predictor t on the perturbed sample.
end
```

With all errors calculated with the pseudo code described in Algorithm 3, the VI coefficient must be calculated using Equation (13) [24].

$$VI(X^j) = \frac{1}{ntree} \sum_{t=0}^{j} (errO\tilde{O}B_t^j - errOOB_t) \qquad (13)$$

Since all VI have been calculated, now it is possible to select the most important variables from the original data set. To perform this action, the steps described in Algorithm 4 must be followed [23].

Algorithm 4: Pseudocode of the variable selection with RF

```
while preliminary elimination and ranking don't finish do
    Sort the variables in decreasing order of RF scores of
      importance;
    Cancel the variables of small importance;
    Denote by m the number of remaining variables.
end
while variable selection doesn't finish do
    Construct the nested collection of RF models
      involving the first k variables, for k = 1 to m, and
      select the variables involved in the model leading to
      the smallest OBB error;
end
```

In the case of this paper, the RF algorithm was applied in the database and Fig. 5 shows the graph with all proteins ordered by the VI coefficient.

4. Experiments

4.1 Database

The database used in the development of this work was the same used by Gòmez et al. and Dantas in their publications. It has values of 120 proteins found by analysis of blood samples from different patients. The ultimate goal of the database is to classify whether a patient can be diagnosed or not with AD or MCI [7] [10].

In his work, Gòmez et al. subdivided the database in 2 sets. The first set contained the results of blood samples of 83 patients.

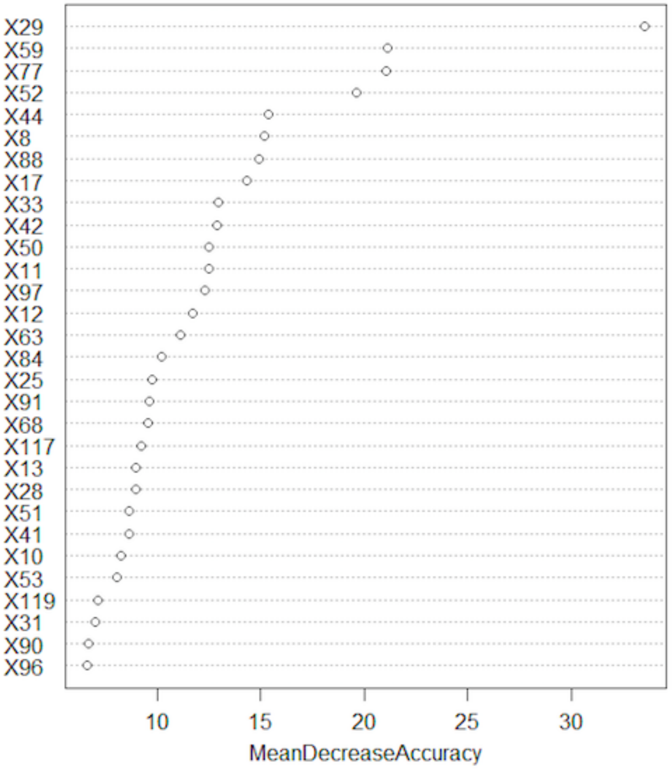

Figure 5. *Application of the RF algorithm in the database used in this work*

Of these 83 patients, 68 were allocated to the training process of the chosen classifier. The data for the remaining 15 patients were used in the process of cross-validation of the classifier, i.e., a process that determines the optimal point to stop its training [11].

The second set, used in the testing process of the classifier, has two options. The first option is it could be used to diagnosis AD and in this case, this set will contain the samples related to the 92 patients that could be diagnosed with AD. The second option is use this set to perform diagnosis of MCI. In this other case, the test set will contain blood samples related to 47 patients with a possible diagnosis of MCI.

Recently, Dantas and Valença [8] [10] defined two new signatures with 10 and 5 proteins using the Random Forest Algorithm. To verify the accuracy of the signatures, they were tested using 3 topologies of neural networks. After all statistical tests, it was concluded that these two new proposed signatures reached better results and the best neural network to perform the diagnosis was the MLP.

In this work, it was used the Random Forest Algorithm to create two new signatures, one for the AD prediction and the other for the MCI prediction, both with 3 proteins.

The Random Forest algorithm was executed 30 times and in each of the simulations, a signature of 3 proteins was defined. After all these 30 simulations were over, the best signature was chosen according to the gini metric.

Table 1 shows the signatures of proteins that are contained in Gòmez et al. work and which are used in this study.

In this work, 8 databases were prepared in order to reproduce the experiments described by Gòmez et al. [7] and Dantas [1], using the MLP, RC and the ELM. They were:

- 1 database for testing the signature of 10 proteins defined by Gòmez et al. [7] with the AD set of tests, called now by **Database 1**;

- 1 database for testing the signature of 10 proteins defined by Gòmez et al. [7] with the MCI set of tests, called now by **Database 2**;

- 1 database for testing the signature of 10 proteins defined by Dantas [1]with the AD set of tests, called now by **Database 3**;

- 1 database for testing the signature of 10 proteins defined by Dantas [1]with the MCI set of tests, called now by **Database 4**;

- 1 database for testing the signature of 5 proteins defined by Gòmez et al. [7] with the AD set of tests, called now by **Database 5**;

- 1 database for testing the signature of 5 proteins defined by Gòmez et al. [7] with the MCI set of tests, called now by **Database 6**;

- 1 database for testing the signature of 5 proteins defined by Dantas [8] with the AD set of tests, called now by **Database 7**;

- 1 database for testing the signature of 5 proteins defined by Dantas [8] with the MCI set of tests, called now by **Database 8**;

- 1 database for testing the signature of 3 proteins defined by the Random Forest Algorithm with the AD set of tests, called now by **Database 9**;

- 1 database for testing the signature of 3 proteins defined by the Random Forest Algorithm with the MCI set of tests, called now by **Database 10**;

All databases described above maintained the organization used by Gòmez et al. regarding the division of values for the training, cross validation and testing set.

4.1.1 Pre-processing of data

To properly execute the training of the neural network, it is necessary that your data is normalized, i.e., the input values of the neural network must be contained in the same numerical range. This is important since very different values can influence the training and generate a loss in the generalization ability of the neural network [11].

Table 1. *Representation of the proteins contained in each one of the signatures used.*

Abbreviation	Signature	Proteins
S1	10 proteins signature defined by Gòmez et al. [7]	CCL7/MCP-3, CCL15/MIP-1d, EGF, G-CSF, IL-1a, IL-3, IL-6, IL-11, PDGF-BB, TNF-a
S2	10 proteins signature defined by Dantas [1] for AD test set	IL-1a, TNF-a, G-CSF, PDGF-BB, IGFBP-6, M-CSF, EGF, IL-3, GDNF, Eotaxin-3
S3	10 proteins signature defined by Dantas [1] for MCI test set	IL-1a, PDGF-BB, EGF, TNF-a, RANTES, FAS, GCSF, MIP-1d, FGF-6, IL-11
S4	5 proteins signature defined by Gòmez et al. [7]	EGF, G-CSF, IL-1a, IL-3, TNF-a
S5	5 proteins signature defined by Dantas using the Random Forest [8] for AD test set	IL-1a, TNF-a, G-CSF, PDGF-BB, M-CSF
S6	5 proteins signature defined by Dantas using the Random Forest [8] for MCI test set	IL-1a, PDGF-BB, EGF, TNF-a, RANTES
S7	3 proteins signature defined by the Random Forest for AD test set	IL-1a, TNF-a, G-CSF
S8	3 proteins signature defined by the Random Forest for MCI test set	IL-1a, PDGF-BB, EGF

One of the most commonly used normalization techniques in literature is the linear transformation and it was the one chosen for this work. Equation (14) is the formula used to normalize the values of the database.

$$y = ((b-a) \times (\frac{x - x_{min}}{x_{max} - x_{min}})) + a \tag{14}$$

In (14), a and b represent the maximum and minimum values that the data should take. In this work, it was used the value of -0.85 for a and 0.85 for b, as the activation function chosen for this neural network is the Hyperbolic Tangent. Therefore, the values contained in the database must be between -1 and 1.

4.2 Simulations

4.2.1 MLP Parameters

Although it is widely used in many researches, several parameters of the MLP must to be configured. The choice of each one directly influences the final outcome of the prediction.

Below are the main parameters of the MLP and Backpropagation algorithm:

- Number of neurons in the input layer;
- Number of neurons in the hidden layer (only one hidden layer);
- Number of neurons in the output layer;
- Activation function of the hidden layer;
- Activation function of the output layer;
- Learning Rate;
- Momentum.

During the experiments, the number of neurons in the input layer was varied according to the amount of proteins in the signature of each experiment. These values can be either 5 or 10.

This work aimed to perform early diagnosis of a patient with or without AD or MCI, thus the number of neurons in the output layer is 2. The algorithm used to adjust the weights is the Backpropagation and its formula can be found in section 2.1.1.

The activation function chosen of the hidden layer is the hyperbolic tangent. This function returns values in the interval [-1, 1] and is given by 11. Several tests were performed to define the learning rate, momentum and number of neurons in the hidden layer. The best results correspond to the values of 0.3 for the learning rate, 0.6 for the momentum and 20 neurons in the hidden layer.

The MLP used has been implemented in the JAVA programming language and in the Eclipse development environment [25].

Table 2. *Representation of the parameters used for the simulations with the RC, MLP and ELM.*

Parameters	RC value	MLP value	ELM value
RC connectivity	10%	Not applicable	Not applicable
Number of neurons in the input layer	Depends of the amount of proteins in the signature	Depends of the amount of proteins in the signature	Depends of the amount of proteins in the signature
Number of neurons in the RC	4	Not applicable.	Not applicable
Number of neurons in the hidden layer	Not applicable	20	20
Number of neurons in the output layer	2	2	2
Number of warm up cycles	100	Not applicable.	Not applicable
Activation function of neurons in the reservoir or hidden layer	Hyperbolic Tangent	Hyperbolic Tangent	Hyperbolic Tangent
Activation function of neurons in the output layer	Linear	Linear	Linear
Learning rate	Not applicable.	0.3	Not applicable
Momentum	Not applicable.	0.6	Not applicable

4.2.2 RC Parameters

As the MLP, the Reservoir Computing technique has several parameters that require configuration. Taking into account that it is a recent area of research, the choice of these settings can not be considered ideal and it is often performed randomly. One way to do this is to evaluate each chosen parameter value and determine if it was better or worse for the network performance. This process is repeated until a value is considered optimal, which does not necessarily means the best.

In this work, these parameter values were changed and all the classification rates at the end of the trainings were stored and then compared. The configuration which presented the best parameter values for the network performance was the one chosen.

Below are the parameters whose settings were required to be defined during this project:

- Number of neurons in the input layer;
- Number of neurons in the output layer;
- Number of neurons in the reservoir;
- Activation function of the reservoir;
- Activation function of the output layer;
- Initialization of weights;
- Connection rate of the reservoir;
- Number of warm up cycles;

The number of inputs and outputs remain the same as used in the MLP, since the goal is to compare the ANN's techniques using the same classification scenario. It means the number of neurons in the input layer can be 5 or 10, depending on the signature used. The number of neurons in output layer is 2.

The number of neurons in the reservoir is one of the parameters for which there is no fixed criterion that defines it. It was chosen randomly after checking the classification rate at the end of each training. It was observed that the ideal number of neurons in the reservoir was 4.

As mentioned in Section 2.3.1, the weights of the input layer to the reservoir and the weights of the reservoir are randomly generated with values between [-1, 1].

The reservoir states are initialized to zero (0). Due to this, as also mentioned in Section II-C, it was decided to add to the network a phase called warm up. During the warm up, it is not necessary to calculate the weights of the output layer, or to calculate an output value. This warm up phase is done just to update the states of the reservoir and remove the dependence of the initial state. The number of cycles chosen for warm up was 100.

The connection rate of the reservoir neurons was 10%. That is, only 10 % of the connections have weight values different from zero associated to them.

The activation function chosen in the reservoir was the hyperbolic tangent. In the output layer, the selected function was linear one.

During this work, we implemented a Neural Network with the technique of RC in the programming language Java and the Eclipse development environment.

4.2.3 ELM Parameters

Finally, the ELM also needs to have its parameters adjusted. Below are the parameters whose settings were required to be defined during this project:

- Number of neurons in the input layer;
- Number of neurons in the hidden layer (only one hidden layer);
- Number of neurons in the output layer;
- Activation function of the hidden layer;
- Activation function of the output layer;

The same number of neurons in the input and output layer used in the two other techniques is maintained for the ELM. That is, 5 or 10 inputs and 2 outputs.

The number of neurons in the hidden layer remains the same used with the MLP which means 20 neurons.

The activation function of the hidden layer is the hyperbolic tangent and of the output layer is the linear.

Table 2 summarizes the parameters used in all experiments with the neural networks topologies mentioned above.

After defining the settings of the MLP, ELM and RC, 30 simulations were performed with each of the databases in each of the chosen neural network topologies in this work. This number is considered ideal to perform more meaningful statistical comparisons [26].

4.3 Statistical Analysis

When all the simulations were completed, it was necessary to perform a sequence of statistical tests in order to scientifically validate the results. For this, it was used the R mathematical software, since it contains all the implementations of the tests used. This software uses as default a level of significance (α) previously defined with the value of 0.05.

Before using a parametric test on a data set it is necessary to check whether the samples are normally distributed and if they have statistically equal variances. If these two assumptions are validated, one can apply a parametric test, otherwise it must be used a non-parametric test.

In order to verify the first assumption, that is, check if the samples were normally distributed, the Shapiro-Wilk test was applied with the hypothesis described in Table 3.

After that, it is necessary to verify whether or not the samples were drawn from the same population, i.e., if their variances were statistically equal. The hypothesis for this test are available on Table 4.

If these two premises are true, a parametric test can be used. In this case, it was chosen the Student's T-test. The hypothesis are described in Table 5.

In case of the first two premises aren't true at the same time, it is necessary to use a non-parametric test, i.e., one that makes no assumptions about the probability distribution of the samples. It was chosen the Wilcoxon Rank-Sum Test with the hypothesis in Table 6.

For all those cases, the p-value must be compared with the significance level adopted in the R software (α) . In case of this value is less then α the null hypothesis should be rejected implying that the alternative hypothesis is automatically accepted.

Table 3. *Null and alternative hypothesis for the Shapiro-Wilk Test.*

Hypothesis	Description
Null Hypothesis	The sample is normally distributed
Alternative Hypothesis	The sample isn't normally distributed

Table 4. *Null and alternative hypothesis for the F Test.*

Hypothesis	Description
Null Hypothesis	The samples have variances statistically equal
Alternative Hypothesis	The samples don't have variances statistically equal

Table 5. Null and alternative hypothesis for the Student's T-Test

Hypothesis	Description
Null Hypothesis	The average of the analyzed samples are statistically equal
Alternative Hypothesis	The average of the analyzed samples aren't statistically equal

Table 6. *Null and alternative hypothesis for the Wilcoxon Rank-Sum Test*

Hypothesis	Description
Null Hypothesis	The median of the analyzed samples are statistically equal
Alternative Hypothesis	The median of the analyzed samples aren't statistically equal

Table 7. *Representation of the average accuracy rates after the 30 experiments.*

Database	Average classification rate with RC / Standard Deviation	Average classification rate with MLP / Standard Deviation	Average classification rate with ELM / Standard Deviation
Database 1	86.62% / 0.026	93.44% / 0.017	87.78% / 0.025
Database 2	69.29% / 0.024	68.15% / 0.018	68.45% / 0.027
Database 3	90.57% / 0.022	94.31% / 0.008	91.05% / 0.023
Database 4	76.59% / 0.047	78.86% / 0.031	74.32% / 0.050
Database 5	93.22% / 0.026	95.61% / 0.017	91.12% / 0.021
Database 6	65.44% / 0.024	69.14% / 0.018	66.81% / 0.019
Database 7	92.97% / 0.022	93.26% / 0.008	91.62% / 0.028
Database 8	72.90% / 0.047	73.33% / 0.031	73.88% / 0.040
Database 9	92.76% / 0.037	83,70% / 0.013	92.58% / 0.037
Database 10	65.51% / 0.032	72.12% / 0.012	63.76% / 0.028

5. Results

After all simulations were performed with the databases, it was calculated the arithmetic mean for each set of simulations and Table 7 displays those values found.

As none of the ten datasets met the two premises necessary for the application of a parametric test at the same time, it was not possible to perform the Student's T-test. And so, the Wilcoxon Rank-Sum Test was chosen, since it is a non-parametric test, i.e., it makes no assumptions about the probability distribution of the samples.

When applied the Wilcoxon Rank-Sum Test for each of the eight cases, the results found were that the MLP has a statistically better performance than the RC and ELM, except for the Database 9, which was proved that the RC had a better performance for the new signature with 3 proteins and AD test set.

In order to verify the performance of the new proposed signatures, simulations with RC, MLP and ELM topologies were performed for both the diagnosis of AD and MCI. The results were compared with those found by the same neural networks when the signatures used were the ones defined by Gòmez et al and by Dantas and Valença [8] [10].

For the new signature with 3 proteins and the AD test set, the Wilcoxon Rank-Sum test proved that the result with this signature was statistically the same of the one found with the 5 proteins signature created with the Random Forest Algorithm. This is an interesting result, because it was found the same classification rate with a lower number of proteins and then, the cost of this exam will be less expensive.

For the signature proposed with 3 proteins and the MCI test set, the result was worse then the one found with the 5 proteins signature created by Dantas and Valença in [8]. This statement was proved statistically when the Wilcoxon Rank-Sum test was applied.

Table 8 summarizes the maximum values found in the simulations of the RC, MLP and ELM. Those results were found using all new signatures proposed. Table 8 also displays the results obtained in the work of Gòmez et al. using their own signature [7].

Table 8. *Comparison of the maximum values found with the simulations with the new protein signatures proposed obtained with RC, MLP, ELM and the ones available in literature.*

Protein Signature	RC	MLP	ELM	Results found by Gòmez et al.
S2	96.73%	95.65%	96.81%	93%
S3	89.36%	82.97%	89.20%	66%
S5	98.15%	97.33%	95.29%	93%
S6	73.50%	74.21%	74.49%	66%
S7	96.73%	84.44%	95.21 %	93%
S8	70.21%	72.72%	69.67%	66%

From Table 8, it can be concluded that all neural network topologies used obtained results consistent with those described by Gòmez et al. and succeeded in reaching a maximum value greater than the average found in the literature.

6. Conclusion

Nowadays, Alzheimer's disease is one of the most common diseases in the elderly population. In recent years, the number of patients has grown significantly since the life expectancy in most developed countries has increased.

AD is a degenerative disease, i.e., brain cells will deteriorate and there is no way to reverse the disease. However, the earlier the drugs are administered, the better the quality of life of the patient since the medication will slow the progression of the symptoms.

Thus, this study aimed to verify the performance of MLP, RC and ELM to early classify if a patient can be diagnosed with AD or not. Moreover, another goal was to make a comparison of the performance of the MLP with the RC and ELM neural networks, and also with the results available in the literature.

From the statistical tests and simulations, it can be concluded that the MLP presented a superior performance in all cases, except in the case of the signature with 3 proteins and the AD test set, which the RC presented a better performance. It is also possible to conclude that the signature with 3 proteins had the same results with the one with 5 proteins, therefore, this new signature proposed can be used for the AD diagnosis with a lower cost.

The signature with 3 proteins and MCI test set did not achieve a good result. So, for this situation, the signature with 5 proteins defined with the use of the Random Forest Algorithm should be used to perform the early diagnosis of MCI disease.

As future work, it is intended to use other neural networks topologies to make a comparison with those already used in this work. Besides that, it is intended to invest in more variable selection techniques in order to further optimize the results and to reduce the number of proteins in the signatures used to perform early diagnosis of Alzheimer and MCI.

References

[1] DANTAS, Lara et VALENCA, Meuser. A comparative study of neural network techniques to perform early diagnosis of alzheimer's disease, In: *The Sixth International Conference on Advanced Cognitive Technologies and Applications*, 2014, pp. 178-183.

[2] H. Portal. Portal of the brazilian health. [Online]. Available: `http://portal.saude.gov.br/saude/` [Accessed 23 7 2014]

[3] GREEN, R. Diagnosis and Treatment of Alzheimer's Disease and Other *Dementias. Editora de Publicações Científicas Ltda.*, 2001

[4] MARQUES, P. Dementia of Alzheimer type: diagnostic, treatment and social aspects. *Editora de Publicações Científicas Ltda*., 1997.

[5] GIUSTI, E. et SURDO, V. Alzheimer's care and family counseling: psychological needs and treatment of dementia. *Gryphus*, 2010.

[6] RAY, Sandip, BRITSCHGI, Markus, HERBERT, Charles, et al. Classification and prediction of clinical Alzheimer's diagnosis based on plasma signaling proteins. *Nature medicine*, 2007, vol. 13, no 11, p. 1359-1362.

[7] GOMEZ RAVETTI, Martín et MOSCATO, Pablo. Identification of a 5-protein biomarker molecular signature for predicting Alzheimer's disease. *PloS one*, 2008, vol. 3, no 9, p. e3111.

[8] DANTAS, Lara et VALENCA, Meuser. Using Neural Networks and Feature Selection Algorithms in the Identification of Protein Signatures for the Prediction of Alzheimer's Disease, International Journal On Advances in Life Sciences, 2014, 6, no 34, p. 373-382.

SIEVERINK, Floor, BRAAKMAN-JANSEN, Louise, ROELOFSEN, Yvonne, et al. The Diffusion of a Personal Health Record for Patients with Type 2 Diabetes Mellitus in Primary Care. *International journal on advances in life sciences*, 2014, vol. 6, no 3&4, p. 177-183.

[9] HUANG, Guang-Bin, ZHU, Qin-Yu, et SIEW, Chee-Kheong. Extreme learning machine: a new learning scheme of feedforward neural networks. In : *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*. IEEE, 2004. p. 985-990.

[10] DANTAS, Lara et VALENCA, Meuser. Using Neural Networks in the Identification of Signatures for Prediction of Alzheimer's Disease. In : *Tools with Artificial Intelligence (ICTAI), 2014 IEEE 26th International Conference on*. IEEE, 2014. p. 238-242. MLA

[11] M. Valença, Fundamentals of Neural Networks: Examples in Java, 2nd ed. Livro Rápido, 2009.

[12] VERSTRAETEN, David. *Reservoir computing: computation with dynamical systems*. 2009. Thèse de doctorat. Ghent University.

[13] WERBOS, Paul. Beyond regression: New tools for prediction and analysis in the behavioral sciences. 1974.

[14] HAYKIN, S. Neural Networks: Principles and Practices. Bookman, 2007.

[15] RAJESH, R. et PRAKASH, J. Siva. Extreme learning machines—a review and state-of-the-art. *International journal of wisdom based computing*, 2011, vol. 1, no 1, p. 35-49.

[16] Valença, M. Applying Neural Networks: A Complete Guide. 2005.

[17] MAASS, W. Motivation, theory, and applications of liquid state machines. *Computability in Context: Computation and Logic in the Real World*, 2009, p. 275-296.

[18] JAEGER, Herbert. The echo state approach to analysing and trainning recurrent neural networks, *German National Resource Center for Information Technology*, Tech. Rep., 2010.

[19] MASSAR, Marc et MASSAR, Serge. Mean-field theory of echo state networks. *Physical Review E*, 2013, vol. 87, no 4, p. 042809.

[20] Lukoševičius, Mantas. A practical guide to applying echo state networks. In : *Neural Networks: Tricks of the Trade*. Springer Berlin Heidelberg, 2012. p. 659-686.

[21] ARAÚJO, A. *A method for design and training reservoir computing applied to prediction of time series*, Ph.D. dissertation, Universidade de Federal de Pernambuco, 2001.

[22] KULKARNI, Manjari S. et TEUSCHER, Christof. Memristor-based reservoir computing. In : *Nanoscale Architectures (NANOARCH), 2012 IEEE/ACM International Symposium on.* IEEE, 2012. p. 226-232.

[23] LIAW, Andy et WIENER, Matthew. Classification and regression by randomForest. *R news*, 2002, vol. 2, no 3, p. 18-22.

[24] GENUER, Robin, POGGI, Jean-Michel, et TULEAU-MALOT, Christine. Variable selection using random forests. Pattern Recognition Letters, 2010, vol. 31, no 14, p. 2225-2236.

[25] Santos, S. *An application for prediction of water resources based on neural networks and particle swarm*, Master's thesis, Universidade de Pernambuco, 2012.

[26] JURISTO, Natalia et MORENO, Ana M. *Basics of software engineering experimentation*. Springer Science & Business Media, 2013.