Journal of Theoretical and Applied Information Technology
30th November 2021. Vol.99. No 22

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

HYBRIDIZATION OF NEURAL NETWORKS AND GENETIC ALGORITHMS FOR BETTER CLASSIFICATION

[1]MARYEM HOURRI, [2]NOUR EDDINE ALAA
[1]Laboratory LAMAI, Cadi Ayyad University; Morocco
[2]Laboratory LAMAI, Cadi Ayyad University; Morocco
E-mail: [1]maryemhourri@gmail.com, [2]n.alaa@uca.ac.ma

ABSTRACT

In this paper, neural networks and genetic algorithms are used in a hybrid approach in order to increase the quality of classification problems. The proposed method allows us to design the optimal neural network architecture, while avoiding overfitting. In order to evaluate the efficiency of the proposed algorithm, experimental results in three fields are presented by comparing the indices of performance. The hybridization technique produces two desirable effects, a better result and a high performance.

Keywords: *Artificial Intelligence, hybridization, classification, Neural Networks, Genetic Algorithms, Deep Learning.*

1. INTRODUCTION

The rapid development of deep learning and machine learning techniques has sparked much interest in different applications problems. Artificial Intelligence, specifically Artificial Neural Networks (ANNs) and deep learning is revolutionizing businesses and increasing their competitiveness nationally and internationally. Neural networks help us cluster, predict and classify. They have the ability to group or predict or classify unlabeled data according to similarities, when they have a labeled dataset to train on. As an example of classification with neural networks the reader can consult the work [14]. The application of deep learning for classification and prediction is already mentioned in several studies. However, the deficiencies of deep learning and neural networks reside in the ability to found the number of nodes adequate for each layer in order to have a high performance: a good classification or prediction. Our current study plans to implement a predictive model based on improved deep learning with a method of optimizing the various parameters with genetic algorithms for better prediction.

In order to have a prediction with a high degree of precision, we must be able to combine between an efficient algorithm and good data preprocessing. The major problem that arises in neural algorithms is mainly the number of neurons that must appear in each layer of networks, in order to have a high degree of precision. But the fact of being able to grope between the different possible combinations of the networks is hardly effective. However, because of that we develop an hybrid algorithm that combine deep learning algorithm and genetic algorithms: that can accurately detect the number of neurons, which allows us to have an optimal score. Different methods are presented to maximize and optimize the performance of genetic algorithm like the following works [4] [5].

Our solution will be different, reliable and efficient and will help to make the right choice: the number of neural network in different layers. On set of test and validation, our new model gives very good results in terms of precision and quality of classification, in comparison with other works [5] [6]. In this paper, we propose an "end-to-end" approach for a model of classification or prediction using an algorithm of genetic and neural network algorithm with a large database: thousands of observations, to develop and improve the efficiency of our model.

Our work is organized as follows: In the first section, we develop in detail the different tools on which we will be based to treat our solution: the main components of neural networks and precisely Patternet, as well as those of genetic algorithms. The second section contains our hybridization in detail with the different algorithms used. Our third section will be reserved for the different applications of our work and their results, in order to demonstrate the effectiveness of our method. The

last section is a conclusion and discussion based on the results obtained in the previous section.

2. RELATED WORK

Recently, the design of neural network architectures has largely moved from groping with the number of neurons to a more developed automatic methods, often called Neural Architecture Search (NAS), among its methods we find the following [21, 20, 21, 22, 23, 24]. The majority of this works and methods are based either on basic algorithms through reinforcement learning (RL) [20, 21, 23] or an evolutionary algorithms [19, 24, 22]. There are also standard methods for calculating the number of neurons such as the method of (Heaton, 2005) [25] or that of (Tamura & Tateishi, 1997) [26], these methods allow to calculate the number of neurons in each layer with a single formula so we will have a similar number of neurons in each layer. These methods do not permit having different numbers of neurons in each layer, however, in programmed search there is a several probabilities of possible neuron network architectures.

2.1 Artificial Neural Networks (ANNs)

Neural networks have become an important classification tool, recent research in classification has shown that neural networks are a promising alternative to the conventional classification methods; we can note it in the work [18]. ANNs are the technology that was originally born from the understanding of some aspects of the functioning of biological systems proves it in [13].

A multi-layer network is composed of number of neurons, arranged in levels; all neurons are connected to each other through bonds. The first layer is made up exclusively of inputs and is called the input layer. The last layer represents the classification solution on the name: the output layer. An unlimited number of hidden layers are stored between the input and output layers. Figure 1 below represents an example of neural network architecture.

A multilayer network consists of many neurons, arranged in levels; each level is interconnected with the one above and below. The first layer receives external inputs and is called the input layer. The last layer provides the classification solution and is called the output layer. An unlimited number of hidden layers are sandwiched between the input and output layers It is believed that a three-layer network can accurately classify any non-linear function.

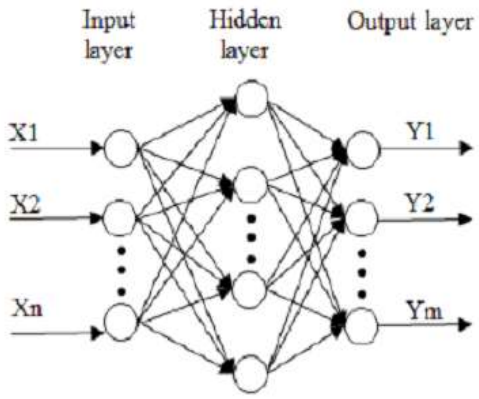

Figure 1: Example of a Neural Network Architecture

The neural systems are organized in layers, the layers are composed of a number of artificial neurons, which contain an activation function.

The patterns are in the network through the input layer, which emits a number of signals to one or more hidden layers, after the treatment is established through a partially weighted connection number. The hidden layers are then linked to an output layer where the response is produced. Standard neural networks contain learning rules which modify the weights of the connections according to the input patterns.

These steps are defined in the works [15] [45].Typically, standard neural networks have the ability to learn, remember and build relationships between the data. Neurons process input data from many modes: first by forming linear combinations of the input Data. Neurons, process the input data in two ways: first by forming linear combinations of the input data, and then by squashing these linear combinations via the activation function, usually the sigmoid function. The sigmoid function is represented by:

$$S(t) = \frac{1}{1+e^{-t}} \qquad (1)$$

To increase the invariance of the model with respect to the output of the classifier we use the softmax activation:

$$f(x_j) = \frac{e^{x_j}}{\sum_{i=1}^{n} e^{x_i}} \quad \text{For j=1,..,n} \quad (2)$$

However, the softmax activation diminishes gradients for large inputs, which may impede gradient flow when it is used in an intermediate layer. Therefore, the rectified linear units (ReLU) can be used instead:

$$f(x_j) = \max(0, x_j) \quad \text{For j=1,...,n} \quad (3)$$

Creating architecture of the neural network comes down to determine: the number of layers, the numbers of neurons in each layer and the choice of activation functions, in order to train the network.

The problem that arises in the construction of the latter is the absence of a general framework to generate an optimal architecture of the neural networks.

Despite the promising results that this method generates in the fields of classification and prediction, there has always been an obstacle, since there was no general framework in order to create the architecture of the neuron networks.

Neural networks have demonstrated their relevance in solving prediction and classification problems. No other technique than neural networking has the capacity to learn from a history, which is important since it detects non-visible patterns.

The problem is that this method encounters is the ability to find the optimal architecture for the network. In this paper, we will perform the neural approach with the genetic algorithm to show the performance of our proposed model, which will allow us to know the number of neurons we need for our model.

2.2 Patternnet

In Matlab the patternet function is used to generate a pattern recognition network. The patternnet is based on a deep artificial neural network to learn how to train different models on complex spaces.

The difference generated with respect to each network is caused by the number of neuron layers and also by the activation functions.

Once the neural network is established the Patternnet incorporations can be used as feature vectors with standard machine learning algorithms.

Patternnet enables a deep neural network to create architecture of a model by drawing inspiration from generative antagonist networks.

The main characteristic of this model is deep learning to constitute the internal architecture of the inputs and to learn the various links between them. Patternnet are networks which are used to classify inputs to have precise output.

The construction of the neuron network as well as its training with patternnet is below:

Algorithm 1: Steps of patternet algorithm:

1: Input the data: x: the attributes and t: the target
2: Choose the architecture of the network: [a1,a2,…,aq], (aj) represent the number of neurons in the jth hidden layer)
3: Construction of the architecture of the classification model: net = patternet([a1,a2,…,aq)]
4: Train the model: net= trai (net,x,t)
5: Generation of the simulate target y= net(x)

2.3 Tamura and Tateishi method for determining number of hidden neurons: (CAT)

The method of Tamura and Tateishi is a standard method, which allows calculating the number of neurons in each layer in a standard way. According to the proof in the following article [26]: a three-layer feedforward network with Ni-1 hidden units can give exactly any relationship between the input and the target.

The network takes a number of N_h neurons in each layer. The hidden neurons are calculated as follows:

$$N_h = N_i - 1 \quad (4)$$

N_h : Number of hidden neurons
N_i : Number of input neurons

2.4 Basic Genetic Algorithms

A genetic algorithm is an evolutionary optimization algorithm inspired by the genetic domain and natural evolution methods. It is mainly based on the Darwinian metaphor, which is based on the fact that a species are in constant evolution by reproduction and selection in order to maximize their performance in their environment. The GA is based on the fact that a species are in constant evolution by reproduction and selection in order to maximize their performance in their environment, based on four laws:

• The law of growth and reproduction.

• The law of inheritance which almost implies the law of reproduction

• The law of variability, resulting from the conditions of existence.

• The law of multiplication of species which brings about the struggle for existence and which results in natural selection This algorithm solves optimization problems by looking for one or more extremes in a given interval while improving an objective function. Related to [9].

The primordial steps will serve us as a basis for applying our principle of genetic algorithm is the modeling of the problem in the form of manipulable computer data.

The Genetic Algorithm is an algorithm which is based on selective genetic research for the survival of the fittest among the chain combinations, in order to have the best possible results.

GA is mainly used for optimization problems with several parameters and an objective function. The principle of genetic algorithms is to start with already existing solutions to the problem and then try to produce new generated solution, that are better than the previous ones, based on techniques derived from genetics and natural evolution: crosses, mutations, selection.

These algorithms use a vocabulary similar to that of genetics.

To optimize a problem using GA, it is first necessary to have already predefined solutions of the problem, in order to: generate an initial population of solutions, an evaluation function that represents the characteristics of individuals and new genetic operators, e.g. population size, number of generations and probabilities of application of genetic operators.

As a starting point for the research, an initialization of the chromosomes (a population of genetic parameters) will be randomly selected in the solution space.

These chromosomes will subsequently be encoded by binary or other chains. Second, each chromosome is evaluated using a predefined fitness function. The interest of this fitness function is to evaluate the performance of each chromosome.

The algorithm of simple genetic algorithm is below:

• Generating an initial population.

• Evaluating the chromosomes in the population.

• Preferentially transferring the fit test chromosomes to the next generation.

• Applying cross over and mutation function for creating new chromosomes.

• Repeat until stopping criterion is meeting.

• Return the best chromosome as the solution.

Figure 2 below represents an example of genetic algorithms architecture.

Through the combination and hybridization of neural networks and genetic algorithms, our prediction model will be based on prediction using artificial neuron networks.

However genetic algorithms will also be present in our prediction model, in order to be able to extract the optimal number of neurons for each layer, to have the best possible classification and the lowest error function.

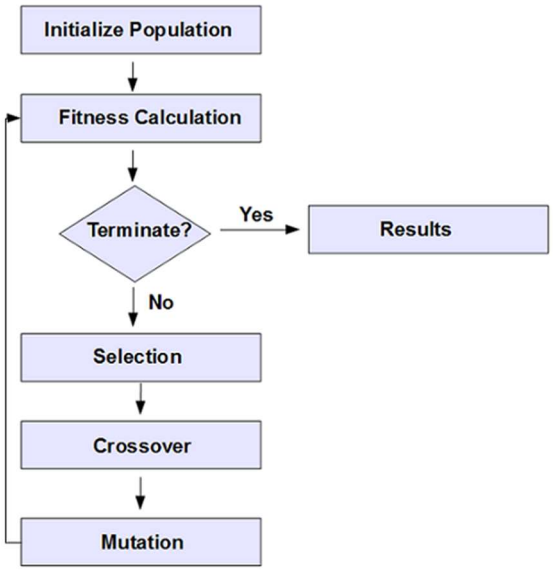

Figure 2: Basic Structure of Genetic Algorithms

2.5 Classifier performance evaluation:

In order to judge the performance of our work we will need the performance indices: accuracy, sensitivity, specificity.

We extract the different performance indices from the confusion matrix, which helps to evaluate the

performance of machine learning model for classification problem.

The accuracy of a test is its ability to differentiate the patient and healthy cases correctly. To estimate the accuracy of a test, we should calculate the proportion of true positive and true negative in all evaluated cases.

$$\text{Accuracy} = \frac{(TP+TN)}{(TP+FN+FP+TN)} \quad (5)$$

The sensitivity of a test is its ability to determine the patient cases correctly. To estimate it, we should calculate the proportion of true positive in patient cases.

$$\text{Sensitivity} = \frac{TP}{(TP+FN)} \quad (6)$$

The specificity of a test is its ability to determine the healthy cases correctly. To estimate it, we should calculate the proportion of true negative in healthy cases.

$$\text{Specificity} = \frac{TN}{(FP+T\ )} \quad (7)$$

Where TP indicates true positive, TN indicates true negative, FN indicates false negative, and FP indicates false positive.

3. FEATURE CLASSIFICATION USING NEURAL NETWORKS AND GENETIC ALGORITHMS (HA)

3.1 Hybrid Genetic Algorithms Deep Learning

The hybrid approach combines the methods of genetic algorithms and neural networks to have an efficient classification and prediction system even recognizing them for quite complex classification problems, its benefits are in terms of research process as well as efficiency of performance.

The advantage of the hybrid approach is the ability to generate higher quality relationships compared to other learning methods in terms of performance and efficiency. neural networks are used to model the relationships between parameters, while genetic algorithms are applied to find an optimal architecture for the network, the combination and hybridization of the 2 models has been proven to deal with complex scheduling problems.

The goal of our hybridization of neural networks and genetic algorithms is to be able to make a relevant classification or prediction. Knowing already that neural networks are deep learning methods that have their relevance, we will add to that the algorithms genetics with its different functions in order to be able to optimize the parameters of neural networks more precisely, optimization the numbers of neurons in hidden layers, to ultimately have maximum precision in our prediction or classification, which is the main goal of each algorithm of deep learning.

We noted that the importance of the number of neurons in each layer comes from the fact that these neurons treat each part of the information that is transmitted. Through the combination and hybridization of neural networks and genetic algorithms, our prediction model will be based on prediction using artificial neuron networks; however genetic algorithms will also be present in our prediction model, in order to be able to extract the optimal number of neurons for each layer, to have the best possible classification.

3.2 Structural and Architecture Design of NN with GA

At this stage of our study we are going to proceed to the hybridization through this algorithm, for each example in the training data list, give input and the output to the model.

• **Initialization**: For each individual we set the number of hidden layers N, which we want in our deep learning model. To manage an initial population: the number of neurons in each hidden layer.

We generate randomly by using the uniform low, N individuals:

$$[a_1\ a_2, \dots, a_q] \quad 1 \le a_i \le L$$

a_i: represent the number of neurons in the ith hidden layer.

The fitness function in our case is the score that is generated by the deep learning.

The chromosome is the number of neurons in each hidden layer and the individual is the architecture of the Deep Learning: $[a_1\ a_2, \dots, a_q]$

The fitness function F is defined by:

$$F([a_1 a_2, \dots, a_q]) = \frac{1}{N}\sum_{i=1}^{n}(y_i - t_i)^2 \quad (8)$$

y_i: Predicted output by the algorithm 1

t_i : The desired target

N: The number of individuals

a_i : Represent the number of neurons in the ith hidden layer

• **Selection:** In this step to improve the overall physical condition of populations, we will make a selection that will be done by choosing the individuals who will have a high score function compared to the others, by rejecting bad conceptions, as we will keep only the best number of neurons in each hidden layer that will help us to have the higher score. We used the selection of the work bellow [10].

• **Crossover:** The previous operation searches for the best individuals, but does not produce new chromosomes.

For this, we will proceed to this crossover step, which will generate us a new integer combinations of numbers of neurons in each hidden layer, through the crossing of two parents P1 and P2, subsequently we will have two new child chromosomes C1 and C2 generated by the parents P1 and P2 In practice, we block a probability Pc, and then we draw a pair [P1;P2] with a random probability u = rand(1,1).

We then start by testing if u > Pc, in this case we go to crossover and we generate the new pair [C1;C2] which replaces the old one, otherwise, [P1;P2] will be kept in the new population. Take for example the following two parents P1 and P2.

1	3	2	5	4	6
9	7	8	6	4	5

After applying the crossover operation with a = 1/3, we obtain the following children C1 and C2:

9	7	2	5	4	6
1	3	8	6	4	5

• **Mutation:** The ultimate goal of this operation is to be able to create diversity in the population and that by replacing some genes of the chromosome by the other allele of the gene, which brings the algorithm to find the optimal solution.
The mutation operator will be used on other P elements with the conversation Pm and the mutated individuals generated are noted by P0.

With some small probability P_m, which decreases over time, each element or coefficient of the two children's vectors is subjected to a mutation.
The probability of each element is subject to mutation in generation G = 1,2,...,Nbgen, given by the probability:

$$P_m = 0.15 + \frac{0.33}{G} \quad (9)$$

The mutation operators usually consist of randomly draw two genes from the chromosome and permute them.

Take for example the following chromosome, after choosing randomly two genes (4 and 7 in this example), we permute them:

3	7	5	2	4	9
3	4	5	2	7	9

And this by using the probabilityP_m, on an individual fact and a random probability u=rand ().

We test if u < P_m, we apply the mutation and the new individual P0 replaces the old one.

The level of adaptation of children (C1, C2) and mutated individuals P0 is then evaluated before integration into the new population.

Calculate the new output of the model with the new numbers of nodes in the hidden layer.

• Calculate the new score and compare it with other ones.

Propagate layer-to-layer error back propagation with delta rules to update the weights of hidden layers (update all network).

• Repeat steps until desired optimal score is reached.

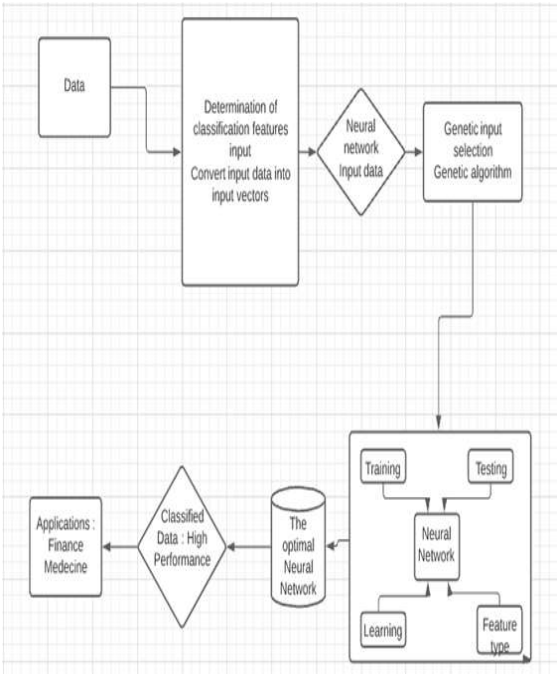

Figure 3: Feature Classification System Architecture

Algorithm 2: Steps of the used algorithm: Algorithm of classification using Neural Network and Genetic Algorithms

1: Generate randomly the initial population;

N individuals [a1,a2,…aq], j=1,…,N, the output is calculated by the neuron network : algorithm 1, a_i : Represent the number of neurons in the ith hidden layer

2: For each generation g

Calculate for each element of the population [a1,a2,…aq] the score by the function means in equation 8

3: Select them using the wheel of fortune; the element with the best score

4: under a probability PC cross the element of the new population using a cross cut (1/3, 2/3)

5: Under a probability pm = 0.13+0. 33/g, mutates the element of the new population?

7: g=g+1, increment and move to the best generation

8: The algorithm stops if the score of the population is exactly equal to the optimal score

9: end for

4. APPLICATION AND IMPLEMENTATION

4.1 Data Pre-processing

Often the raw data is of low quality and this is due to the large size of the databases current.

They can be found incomplete (missing values or aggregated), noisy (erroneous values or aberrant) or inconsistent (divergence between attributes).
The use of data mining on such end data rather complex learning and hurts performance as well than the reliability of the model.

The data pre-processing phase is a very important and often time-consuming phase, due to the lack of structuring and the large amount of noise in the raw data.
Indeed, it improves the quality of data subsequently submitted to data mining algorithms. Pretreatment data is made up of different successive stages:

• Data cleaning: Deleting or correcting records with invalid values in the raw data, and also deleting records for which a large number of columns are missing.

• Setting characteristics: Improved data quality for deep learning, including scaling and normalizing numerical values, introducing missing values, correcting anomalies, and adjusting values with asymmetric distributions.

• Transformation of the representation: Transformation of a numerical characteristic into a categorical characteristic (through the creation of bucket) and transformation of categorical characteristics into numerical characters (one-hot encoding, counting drive, continuous vector representations of hollow features, etc.).

4.2 Setup and Processing of the Dataset: Credit Risk

The database that we used in this study contains 1000 entries with 9 categorical/symbolic attributes and the target. In this database, each entry represents a person who takes a credit by a bank. Each person is classified as good or bad credit risks according to the set of attributes.

The purpose of this study was to predict the credit risk status (good or bad) of each client. We performed an 80% -10% data distribution to create independent training and test sets. In the training set, we also isolated 10% of customers to create an independent validation set.

Journal of Theoretical and Applied Information Technology
30th November 2021. Vol.99. No 22

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

The splits were performed in a stratified manner to maintain the same proportion of default cases in the training, validation and test sets.

The selected attributes are:

- Age (numeric)
- Sex (male or female).
- Number of Job (numeric).
- Hosting (text: Tenant, Home Owner or Other).
- Saving accounts (numeric).
- Checking account (numeric).
- Credit amount (Numeric).
- Duration
- Purpose.
- Status (value target: 1 or 0 (good or bad risk)).

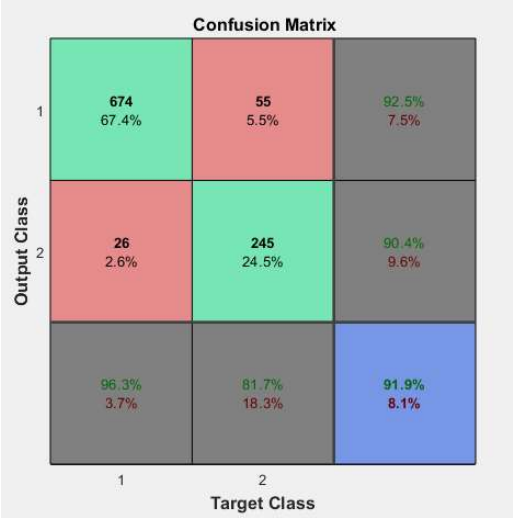

Figure 4: Confusion Matrix Obtained by Our Algorithm: Credit Risk

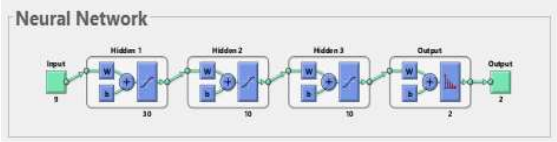

Figure 5: Optimal Architecture

By applying our proposed model: Hybridization of neural networks and genetic algorithms for better classification we obtain satisfactory results (Figure 4).

The confusion matrix shows the percentages of correct and incorrect classifications. Correct classifications are the green squares on the matrix diagonal.

The red squares represent incorrect classifications. After using the hybridization of neural networks and genetic algorithms to train the model, the result of the classification is represented by the confusion matrix.

The blocks in green represent the number and percentage of correct classifications obtained by the network employed.

In our case, 674 customers are correctly out of a total of 1000 customers. Similarly, 245 customers are correctly classified as safe ones.

Overall, Percentage Correct Classification is 96% and Percentage Incorrect Classification 4%.

The accuracy of the model hybrid: 0.96
The sensitivity of the model hybrid: 0.96
The specificity of the model hybrid: 0.81

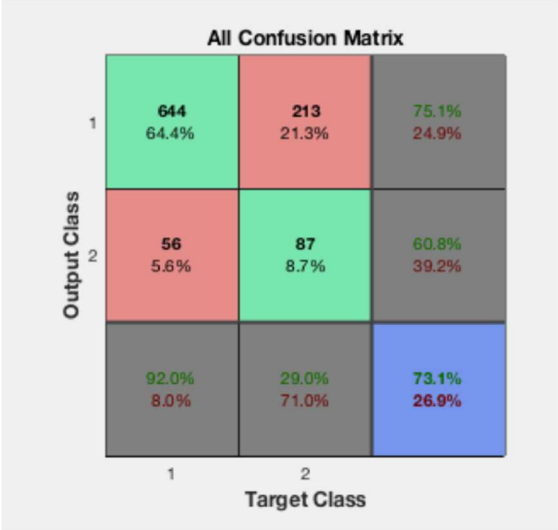

Figure 6: Confusion Matrix Obtained by the Classical Algorithm: Credit Risk

The optimal architecture generated by our algorithm, which allowed us to have this classification is represented in Figure 5.

To prove the utility and the performance of our model it is necessary to compare it with the standard method.

By applying the classical method which is the basic neural network with a random choice of number of neuronusing Tamura and Tateishi method, in this case we used a network of 3 hidden layers:

The input layer contains 8 neurons, the first hidden layer contains 8 neurons, the second one 15 neurons, the third one 8 neurons and the last layer which is the output layer contains 2 neurons.

As a result we have the confusion matrix in Fig. 6 with a 73.1% of the predictions classification that are correctly classified and 26.9% are bad classified.
The accuracy of the standard method: 0.73
The sensitivity of the standard method: 0.92
The specificity of the standard method: 0.29

Our study demonstrates that the choices of the deep learning models trained in an end-to-end are highly accurate and can be easily transferred to banking sector platforms.

Thus, deep learning methods have enormous potential to improve the accuracy of default risk detection as the available training datasets expand. our method work will demonstrate a new risk management approach different from that used in another approach, like the one used in these works [1] [17] [16].

Our work has shown that a simple deep learning algorithm with randomly placed numbers of neurons is not efficient enough, and the choice of several combinations of each number of neurons in each hidden layer; to have the best performance is a waste of time and a risk of not trying all possible combinations.

While our model allows obtaining the best combination of neural networks for each hidden layer for better classification.

4.3 Setup and Processing of the Dataset Cancer:

The data of our second application model is from the FDA-NCI Clinical Proteomics Program Databank. Each column of dataset Cancer represents measurements taken from a patient. There are 216 columns for 650 patients, out of which there is ovarian cancer patients and normal patients.

The model will classify the data into 2 categories: cancer patients and normal patients. Like the problem of this work [3] , but we'll treat it in a different way After the training network with our model the network outputs are in the range 0−1.

The outputs 1 and 0 indicate cancer or normal patients, respectively.

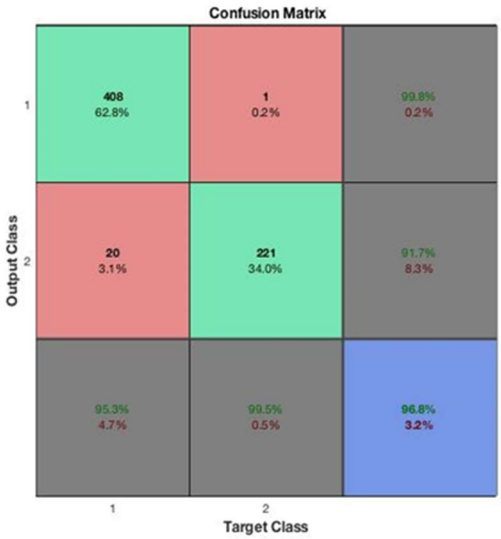

Figure 7: Confusion Matrix Obtained By Our Algorithm: Cancer

After the step of training the result of hybrid classification is shown in confusion matrix in Figure 7.

Percentage Correct Classification: 96.8%.
Percentage Incorrect Classification: 3.2%.
The accuracy of the model hybrid: 0.96
The sensitivity of the model hybrid: 0.95
The specificity of the model hybrid: 0.99

The optimal architecture generated by our algorithm is represented in Figure 8.

In order to do a comparison, the matrix above is the matrix of the classification of patients with cancer with the basic method of neural networks in Figure 9.

Figure 8: Optimal Architecture

Journal of Theoretical and Applied Information Technology
30th November 2021. Vol.99. No 22

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

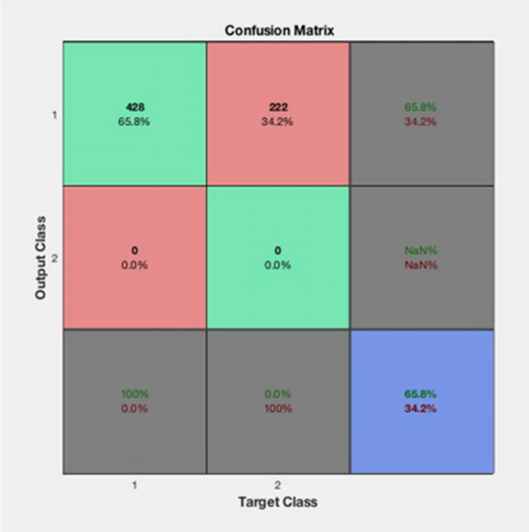

Figure 9: Matrix Confusion Of Classic Deep Learning Cancer

The accuracy of the standard method: 0.65
The sensitivity of the standard method: 1
The specificity of the standard method: 0

This example demonstrate how Hybridization neural networks and genetic algorithms for better classification neural networks can be used as classifiers for cancer detection with high classifier degree and good classification.

4.4 Setup and Processing of the Dataset IRIS:

IRIS dataset consists of 150 data points, 4 inputs and 1 output. The output is the name of flowers and the input is characteristics of the flower. The IRIS dataset was designed to test the accuracy of different classification methods
In this data we attempt to build a neural network that clusters iris flowers into natural classes, such that similar classes are grouped together. Each iris is described by four features: This is an example of a clustering problem, where we would like to group samples into classes based on the similarity between samples.

By applying our Hybridization of neural network and genetic algorithms for better classification, we obtain a good classification. (Fig. 10).
The optimal architecture which allowed us to have this classification is represented in Fig. 11.
We will compare it with the classic method (CAT), in figure 12: we obtain 96.7% of good classification.

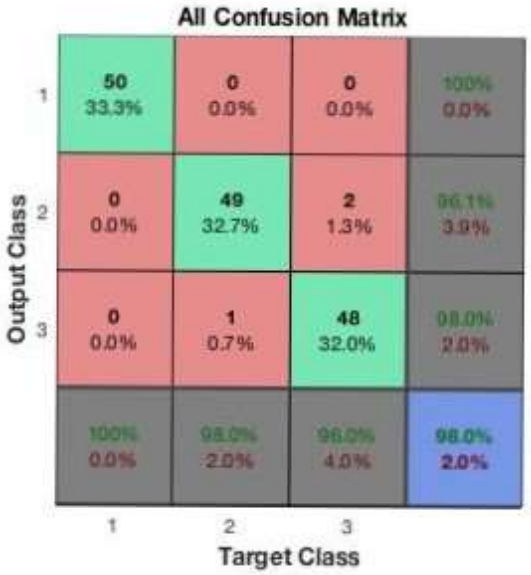

Figure 10: Confusion Matrix Obtained By Our Algorithm: IRIS

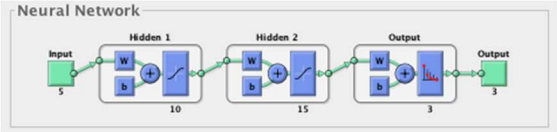

Figure 11: Optimal Architecture For IRIS Model

The accuracy of the model hybrid: 0.98
The sensitivity of the model hybrid: 1
The specificity of the model hybrid: 0.98

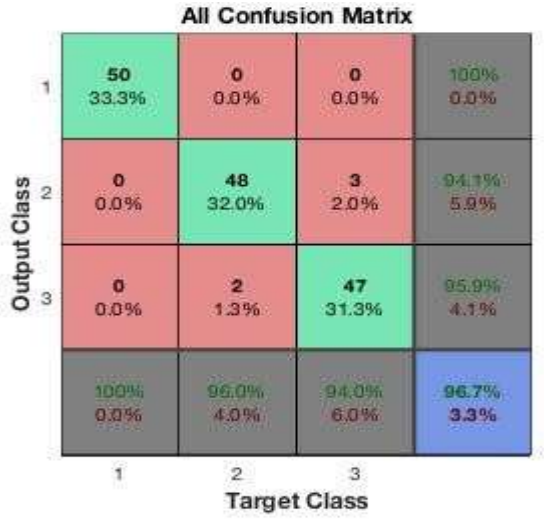

Figure 12: Confusion Matrix Obtained By The Classic Algorithm: IRIS

The accuracy of the classic algorithm: 0.967
The sensitivity of classic algorithm: 1
The specificity of classic algorithm: 0.96

5. DISCUSSION

After the three applications, we notice that the accuracy, the sensitivity and the specificity of the hybridization algorithm between the neural networks and the genetic algorithm are much better than the standard method: of calculating the number of neurons for each layer. The table in the next section summarizes the different results of the 2 methods.

In this study, the databases were reduced to accommodate the available GPU (8 GB).
The following studies if the GPU memory is large will be done on larger databases. Workings on large databases will affect the quality of classification.
In conclusion, our study demonstrates that hybridization allowed us to have optimal neural network architecture for classifications with high quality beyond 90%.

Thus, the hybridization method between networks of neurons and genetic algorithms improves the classification for a better detection, either of people at risk for credit risk according to the first application, or for the detection of cancer patients according to the second application.
Our approach can help banking systems to detect people at risk to increase the number of the accepted credit folders and reduce the rejected ones, or detect cancer patients with better accuracy, while saving time, financial and human resources. This method can be applied on other financial or medical issues.

We will use our approach for applications in medicine, biology, hydrogeology and others.

6. COMPARATIVE ANALYSIS:

This section is reserved for the comparison of the 2 methods used: hybridization and the classic method for the different applications. The table below summarizes the different results obtained for better visibility.

Table 1: Classification of data credit risk

Methods	Accuracy	Sensitivity	Specificity
HA	0.96	0.96	0.81
CAT	0.73	0.92	0.29

Table 2: Classification of data cancer

Methods	Accuracy	Sensitivity	Specificity
HA	0.96	0.95	0.99
CAT	0.65	1	0

Table3: Classification of IRIS DATA

Methods	Accuracy	Sensitivity	Specificity
HA	0.98	1	0.98
CAT	0.967	1	0.96

As shown in the previous tables, the hybrid classifier has high performance compared to the classic classifier: Tamura and Tateishi.
So the optimal architecture obtained by the hybrid model gives very good performing results.

7. COMPUTATIONAL ENVIRONMENT

All experiments in this study were carried out on a windows workstation equipped with an NVIDIA 8GB. The software used is Matlab 2018.

8. APPENDIX

All data used are referenced as follows:
• Credit-Risk:
https://www.kaggle.com/arnavsivaram/credit-riskdata-file
• Cancer:
https://archive.ics.uci.edu/ml/datasets/Breast+Caner
• IRIS:
https://fr.mathworks.com/help/deeplearning/ug/iris-clustering.html

9. CONCLUSION:

In this article, the study has gone from existing methods to determine the number of neurons hidden in each hidden layer of the neural networks to a comparative analysis with the hybridized method between neural networks and genetic algorithms. The comparison was made on three datasets in the field of finance and biology, with a different number of input values. On the basis of the results obtained, it can be concluded that the hybridization method is much better than the classical method for determining the number of neurons in each layer. The hybridized method allows us to choose the final topology of neural networks for a specific problem without resorting to testing the network with a different number of hidden neurons while having high performance.

As future research, there are many works to do. To name a few: an optimization using Particle Swarm Optimization, Grey Wolf Optimizer, Ant Lion Optimizer, Sine Cosine Algorithm, Moth Flame Optimizer.

REFRENCES:

[1] R. Anderson, the Credit Scoring Toolkit, Oxford University Press, Oxford 2007.

[2] T.P. Conrads, et al., quot, " High-resolution serum proteomic features for ovarian detection quot" ;, Endocrine- Related Cancer, pp. 163-178.11, 2004.

[3] K. Deepshikha, "Supervised and Unsupervised Document Classification-A survey", International Journal of Computer Science and Information Technologies, pp. 1971–1974, 2015.

[4] S. Duy Dao, K. Abhary, and R. Marian, " Maximising Performance of Genetic Algorithm Solver in Matlab ". IAENG International Journal of Computer Science, Vol 24, Issue 1, pp 2016.

[5] S. Duy Dao,K. Abhary et R. Marian, " Optimisation of Resource Scheduling in VCIM Systems Using Genetic Algorithm" . IAENG International Journal of Computer Science, Vol. 1 N°8, 2012.

[6] P. O. Duda and P.E. Hart, "Pattern Classification and Science Analysis", Wiley, New York, 1973.

[7] E. Evans" Improved measurements of erythrocyte geometry Microvascular Research ", 4 (4): 335-347, 1972.

[8] E. F. Petricoin, et al., quot; "Use of proteomic patterns in serum to identify ovarian cancer quot", Lancet, 359(9306), pp. 572-577, 2002.

[9] Gabli, Mohammed; Jaara, El Miloud; Mermri, El Bekkaye; "A Genetic Algorithm Approach for an Equitable Treatment of Objective Functions in Multi objective Optimization Problems" , IAENG International Journal of Computer Science, Vol. 41, Issue2 pp. 22-31, 2014.

[10] GolSWDdberg, D. E." Genetic algorithms in search optimization and machine learning" Addison-Wesley Longman Publishing Co., Inc.75 Arlington Street, Suite 300 Boston, MA, United States (412) 1989

[11] E.G. Hinton, S. Osindero and Yee-Whye Teh,A "Fast Learning Algorithm for Deep Belief Nets, In: Neural Comput.", 18.7, pp. 1527-1554, 2006.

[12] Hendry; Rung-Ching Chen; "Using Deep Learning to Predict User Rating on Imbalance Classification Data". IAENG International Journal of Computer Science, Vol. 46, Issue 1,pp. 109-117, 2019

[13] J. H. Holand"Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence". Kluwer Academic Publishers-Plenum Publishers; Kluwer Academic Publishers. 1992.

[14] K. Hornik," Approximation capabilities of multilayer feedforward net-works", Neural Networks, Vol. 4, pp. 251–257, 1991.

[15] K.Hornik, M. Stinchcombe and H. White, "Multilayer feed forward networks are universal approximators" , Neural Networks, Vol. 2, pp. 359–366, 1989.

[16] P.D. McNelis, "Neural networks in Finance: gaining predictive edge in the market", Academic Press 2004.

[17] M. Refaat,"Credit Risk Scorecard: Development and implementation Using SAS", lulu.com, 2011.

[18] G.P. Zhang, "Neural Networks for Classification: A Survey", IEEE Transactions On Systems, Man, And Cybernetics-part C: Applications And Reviews, vol. 30, N. 4, 1958.

[19] L. Xie and A. Yuille. Genetic cnn. In 2017 IEEE International Conference on Computer Vision (ICCV), pages 1388–1397, Oct. 2017

[20] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. arXiv preprint arXiv:1611.01578, 2016.

[21] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, 2018.

[23] Risto Miikkulainen, Jason Liang, Elliot Meyerson, Aditya Rawal, Dan Fink, Olivier Francon, Bala Raju, Hormoz Shahrzad, Arshak Navruzyan, Nigel Duffy, et al. Evolving deep neural networks. arXiv preprint arXiv:1703.00548, 2017.

[23] Han Cai, Tianyao Chen, Weinan Zhang, Yong Yu, and Jun Wang. Reinforcement learning for architecture search by network transformation. arXiv preprint arXiv:1707.04873, 2017.

[24] Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc V Le, and Alexey Kurakin. Large-scale evolution of image classifiers. In International Conference on Machine Learning, pages 2902–2911, 2017.

[25] Heaton, J. (2005). Introduction to Neural Networks with Java, Heaton Research Inc.

[26] Tamura, S., Tateishi, M. (1997). Capabilities of a four-layered feedforward neural network: four layers versus three, Proceedings of the IEEE Transactions on Neural Networks (pp. 251-255)