بسم الله الرحمن الرحيم

"El-Dorra" Neuronal Network Technology, Remind and Optimization.

OMARI Abdallah :
"Study and Research Center Of Bioinformatics "ADRAR
omariabdallah@maktoob.com / BP 488 ADRAR (RP) 01000 ALGERIA.

Note:
This paper is the second parts of research presented in the paper entitled "Development of a new neuronal networks Technology : El-Dorra" accepted in CIMCA'04 Australia, ICBP'04 Sweden, ICIMA'04 China, ANNs'04 Portugal, IKS'04 Turkey, ISNB'04 Holland, Bioinformatics'04 Sweden, MML'04 Italy, SCI'04 USA, CITSA'04 ISAS'04 USA, ICDL'04, .. and others. you can also download the paper from the yahoo group :el-dorra.

What's the type of this paper ?:
Invention: Yes; Study: No; Application: No.

What's new in this paper ?:
1. Tree theorems
2. Algorithm of engendering and counting the hidden layers

Abstract:
The objective of this paper is to determinate algebraically the exact number of hidden layers necessary to solve a problem of N types, and also the number of neurons in each one. We propose here a problem more complicated than the XOR problem named the AXOR and we presente its solution.

Keyword:
El-Dorra, El-Kottabia, El-Rachach, Neuronal Network, Back-propagation, Neither Back Nor Propagation, Artificial intelligent, learning, separated learning.

I- Remind of El-Dorra Technology :

We know that the big problem of the MLP model is to define the number of the hidden layers and the number of the neuron in each layer, this concerning the structure, but concerning the learning the back-propagation algorithm, is not able to make the network learn a big number of classes (in majority not more than 10),the duration of learning takes to much time (between 20 minute and 72 hours) and the error is between 0.4 and 0.01 (bad precision). So in the Technology El-Dorra we tray to solve all this problems.

I-2 The Model :

We propose a new model, that connect each neuron in a layer with the two neurons opposite in the next layer, so if we have **M** neuron in the enter layer named "**The Stone**", we insert a medium layer named "**The Father**" with **K=M+N-1** neurons, so that **N** is the number of neuron in the exit layer named "**The Son**".
Between the **Stone** and the **Father** we insert N-2 hidden layers so the number of neurons in each layer is equal to the number of the neurons in the previous layer +1, and between the **Father** and the **Son** we insert M-2 hidden layer so the number of neuron in each layer is equal to the number of the neurons in the previous layer-1. here there is an example of el-dorra model with M=4 and N=3.

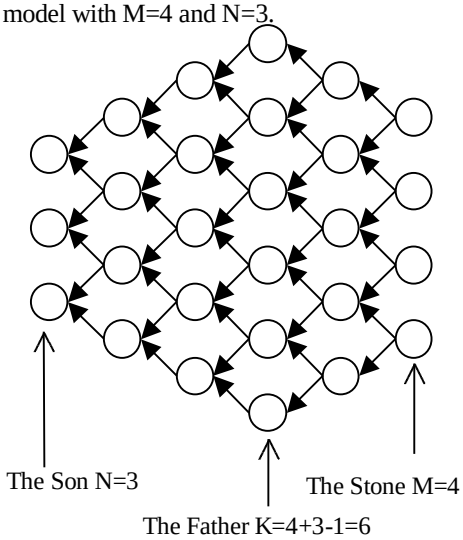


***Fig 1:** Example of El-Dorra model*

we can easily see that this model is structured, the number of hidden layers =**N+M**-3 and the number of neurons is defined in each layer.

I-3 The Learning :

I-3.1 The Problem:
In our study, we have remark that when we use the back-propagation learning algorithm, and enter all the classes together, the network can not be learned easily, but when we made one class alone in the networks, it is learned quickly and precisely.

I-3.2 The Solution:
El-Kottabia Learning Algorithme

I-3.2.1 Definition of El-Kottab :
It is the traditional maghreb school (Algeria, Morocco, Tunisia, Libya, Mauritania, Egypt), the method used in this type of school when we want to learn a text contains many phrases, is to learn every phrase separately, and after we repeat all the phrases together in order to learn (grouped) all the text . so we applied this method on the neuronal networks using the back-propagation in the separated learning.

I-3.2.2(* New)The separated Learning :**
Definition :
We take for every class (type) a separated network (El-Dorra) and we make learning by using the back-propagation, in the end we save the precipitation (sum)or the secretion (exit value) off all the neuron of the network.

Propriety:
In the separated learning the networks is learned in a very short time (0.001 second) and with an very small error (10^{-50}) and by using a few numbers of examples.

This is the first part of the solution, and second is how to make unified the separated neuronal networks of the classes (types) in one???

If we have two separated networks D1 associated to the vector (X1,Y1) and D2 associated to the vector (X2,Y2), so that in every one the two neuron are connected with the opposite neuron (it is the basic principle of connection in El-Dorra), so we can get an unified D that if we enter the first vector (X1,Y1) we get Z1, and if we enter the vector(X2,Y2) we get Z2

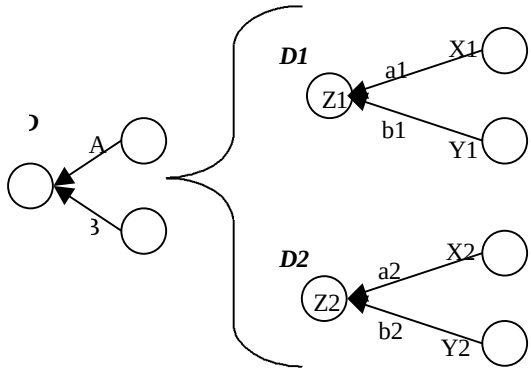


***Fig 2* : Principle of Unification of two Networks**

We can easily solve the system of 2 variables (the new weight of the unified network) and 2 equations :
AX1+BY1=Z1
AX2+BY2=Z2
X1,Y1,Z1,X2,Y2,Z2 are known.
And it is the same thing for 3 or N separated networks, we can easily unified them by solving the system of N equation and N variable deducted from the N separated networks, but in this state we don't need to two entering connectors but we are need to N entering connectors for each neuron.

I-3.2.3 El-Kottabaia learning Algorithm Back-Propagation Version :
Principle :The principle of this version is to make learn every classes (type) alone in a separated network D1,D2,…,Dn by using the back-propagation algorithm, in the end of the learning we save the precipitation of all neurons of the networks.
to unify the networks in one, we make a new networks D that for each neurons there is N entering connectors.
After, for each neuron NR of D we calculate the weight of the N entering connectors by solving the system of N variable :

- the coefficients of the line "i" are the secretions of the neurons connecting to the neuron NR in the separated network Di .
- the right member of the line "i" is the precipitation of the neuron NR in the separated networks Di.

Example :
We take the secretion function f(x)=x, and two **Dorra** Networks D1,D2, so we get the unified network D by the application of El-Kottabia algorithm:

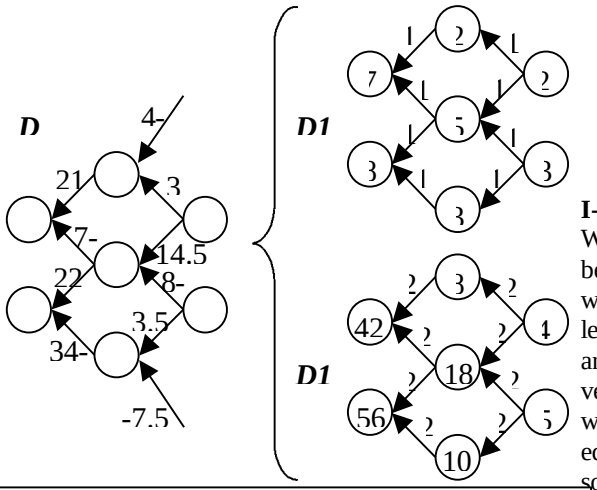


***Fig 3* : Example of Unification of tow Networks**

Notice: The numbers which are on the connectors are their weights.
For example when we apply the algorithm on the second neuron of the first hidden layer we get:
2a+3b=5 (the first network)
4a+5b=18 (the second network)
so a=14.5 and b= -8 are the weight of the entering connectors of this neuron in the unified network.

I-3.2.4 El-Kottabaia learning Algorithm Neither back Nor Propagation version

We remark that in the previous version, the number of the entering connectors in each neuron equal to N (number of types), so in the previous version we search to save the same precipitation of every neuron in the separated network and the unified network, but in this new version we are interested to save the precipitation of the neurons of the exit layer because it is the decision layer.

Algorithm principle

We have N type and M neuron in the entering layer,
We take N separated Dorra networks and we enter every type in one.
Except the exit layer, we calculate the precipitation of the neurons of the N networks.
In the unified network, for every neurons of the exit layer we add a new entering connectors from any neurons (El-Rachache) situated in a previous layers to make the number of the entering connectors equal to N.
For every neuron NR of the ***exit layer*** of the unified network D we calculate the weight of the N entering connectors by solving the system of N variable :

- the coefficients of the line "i" are the secretions of the neurons connecting to the neuron NR in the separated network Di .
- the right member of the lin "i" is the precipitation of the neuron NR in the separated networks Di.

I-4 The learning with error and without error

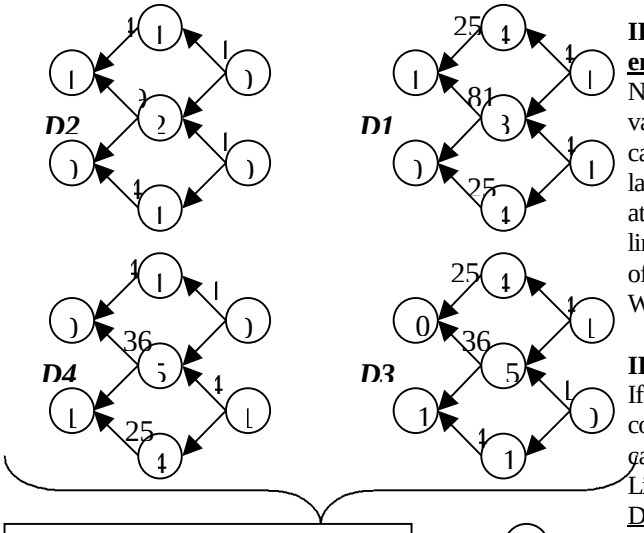
We name the grouped learning, learning with error because the Epsilon is a factor of the algorithm and we search to obtain a weights who makes the error less of this Epsilon, but in the ***El-Kottabia Algorithm*** and exactly in the Neither back Nor propagation version , Epsilon is not a factor of the algorithm and we search to obtain a weights who makes the error equal to zero, and this is guaranteed because the solution of the linear system of N equation is always possible without error (for any N), we say also this type of learning the exact learning, the algebraic learning or the insertion of knowledge.

Problem

The big problem of this method is when the determination of the system equal to zero, this state is marked when the Coefficients (the exit values) are linearly relied, for this case we must use a no linear function to generate a no linear variable.

I-5 The solution to of the XOR problem according to El-Dorra Technology

We take $f(x)=(x+1)^2$, the first neuron in the exit layer represent the "False" class and the second represent the "True" class, there is 4 type (1,1), (0,0),(1,0), (0,1) so we need to 4 entering connectors in the exit layer (The son), the values in the top of neurns are their secretion.



The 1fst neuron (determinant=486)

25a+81b+4c+4d=1	a=-0.89
4a+9b+c+d=1	b=0.111
25a+36b+4c+d=0	c=4.889
4a+36b+c+4d=0	d=-1.333

The 2end neuron (determinant=-486)

81a+25b+4c+4d=0	a=-0.111
9a+4b+c+d=0	b=0.556
36a+4b+4c+=1	c=1.333
36a+25b+c+4d=1	d=-2.556

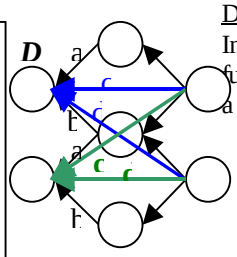


***Fig4* : Solution of the XOR problem**

II-2 Relation between adding a new layer and engender of new variables:
Now, The relation between the engender of the variables and adding a layers are very clear, so we can engender a number of neurons (variables) in a layer from the previous layer, but the question posed at this moment is how many neurons (variables) no linearly relied (NLR) we can engender from a layer of N neurons???!.
We separate here three state :

II-2.1 The First state
If we use a different secretion function for each connector existing from a neuron of a layer L1 we can engender an infinity of neurons (variable) Not Linearly Relied in the next layer L2
Demonstration :
In the universe there is an infinity numbers of NL functions, so we can associate for every NL function a variable(neuron)

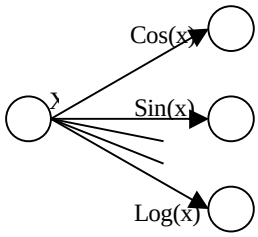


***Fig 5* : Generation of a infinity of neuron from a neuron have a different NL function**

II- Optimization :
We had seen that in the "Neither Back Nor Propagation" version, we must have for each neuron of the exit layer "The son" N enter connectors, but to solve the system of N variable and N equation the Determinant must be different to zero, so all the N variables (secretions) of the system must be no linearly relied

II-1 The First Theorem

To assure the Algebraic learning, the number of the neuron connecting directly to each neuron of the exit layer must be equal to the number of the types and the value of the secretion of this neurons must be No Linearly Relied (NLR. so we can deduct that the number of the neuron in the network (without counting the exit layer) must be equal or bigger than the number of the types.

Demonstration :
To solve the linear system of n equation we must have N variable no linearly relied.

II-2.2 The Second state :
If we use the same no linear function for every neuron, and we use a different weight for the connectors exiting from each neuron of a layer "L1", we can engender an infinity of neurons in the next layer "L2" that his secretion are No linearly relied variables
Demonstration :
We suppose that we can get an limited of number of variable NRL engender from the same variable X (neuron)by using the same function f (x), f(2x), …….. f(I*x), but we can get an other variable f(N+1(x)). so we deduct that the number of variable is not limited.
Example :
Log(x), log(2x), ……, log(I*x)

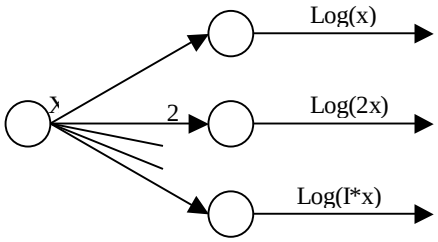


***Fig 6** : Generation of a infinity of neuron from a neuron have a different exit weights*

The Second Theorem

In the algebraic learning, one hidden layer is very sufficient to solve all problems, if we can associate a different functions or different weights far every connector, and more, one neuron in the entering layer is very sufficient to solve all problems under the previous conditions.

Demonstration :
The first and the second state demonstrations are the demonstration of this theorem.

Disadvantages of the two state :
We use a big space to save a different function and the different values of weights.

II-2.3 The Third State:

The Third Theorem

If the value of weight are identical (equal to1) and we use the same NL (No Linear) function for all neurons, so the number of the NLR neurons engendering in the layer "L2" from a layer "L1" containing I neurons is limited = 2^I-1.

Demonstration :
The number of NLR neurons (variable) engendering from I neurons (variable) N1,N2,……,Ni, if we use for all neurons the same NL function and the same weight (1), is the number of partition of the ensemble {N1,N2,…,Ni}= 2^i-1.
Example :
If we have tow neuron in the first layer, we can engender 2^2-1=3 NLR neurons in the next layer when we use the same function and the same weight for all neuron.

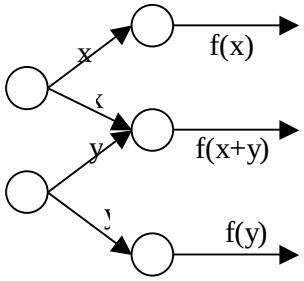


***Fig 7** : engendering of 3 neurons from a layer of 2 neuron*

so, the number of the hidden layers must be existing to solve a problem of N type and M neurons in the entering layer, is :

1. if M=N we connect all the entering layer's neurons with the exit layer's neurons without adding a hidden layers.
2. if M>N, at this moment I don't have a final solution, but we can use the standard El-Dorra model. without optimization or we can add only one hidden layer with N neurons and we connect all the hidden layer's neuron with all the exit layer's neurons, and we connect the entering layer's neurons with the medium layer
3. If M<N we use the following Algorithm (Third theorem)

II-2.3.1 (New) Algorithm of counting and engendering the hidden layers***

1)- Begin
2)- We suppose that :
N: Is the number of type
M: The number of neurons in the entering layer
H: The number of hidden layers
T : The number of neurons in the net (without counting the exit layer)
P : The number of neuron in the previous layer
3)- H=0
T=M
P=M
4)- If T>=N then go to 10
5)- add a layer with 2^P-1 neurons
6)- T=T+2^P-1
7)- P=2^P-1
8)- L=L+1
9)- Go to 4
10)- End.

- we connect the neuron of the exit layer with any N neurons of the net,
- except the previous layer of the exit layer, we connect all parts of the set of neuron in every layer with every neuron of the next layer.

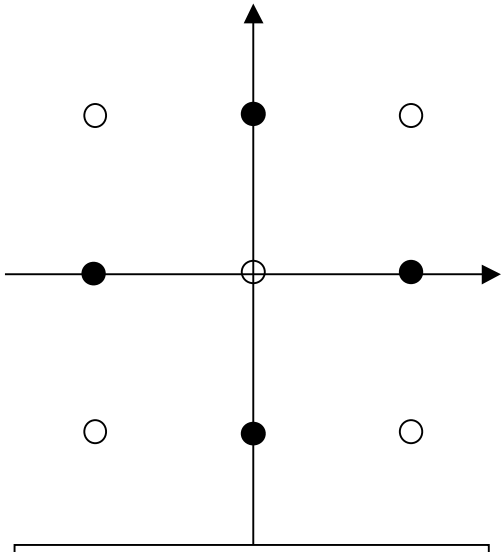
II-2.3.2 <u>The Absolute XOR Problem .</u>

We define the function Absolute XOR (AXOR) by :

AXOR(X,Y)=XOR(ABS(X),ABS(Y)) and the verity table :

X	Y	AXOR(X,Y)
-1	-1	0
-1	0	1
-1	1	0
0	-1	1
0	0	0
0	1	1
1	-1	0
1	0	1
1	1	0

In the plane we can represent this function by:



<u>Fig 8</u>: representation of the AXOR in the space

Notice : This problem can not be solved algebraically using one hidden layer

Because the number of types equal to 9 and the number of the neuron in the entering layer=2<9, so by applying the first theorem we need to add a layer of 2^2-1 =3 neuron but at this time the number of neuron in the network equal to 2+3=5<9, so by applying the first theorem we need to add a hidden layer of 2^3-1=7 and in this time the number of neuron in the network =2+3+7=12>9, so the number of hidden layers must be at least two to solve this problem.

We use the no linear function $f(x)= x^3+2$, the first neuron in the exit layer (the son) represents the "False" class contains 5 type (-1,-1) (-1,1) (0,0) (1,-1) (1,1) and the second neuron in the exit layer (the son) represents the "true" class contains 4 types (-1,0) (0,-1) (0,1) (1,0), all the weights in the networks equal to 1, except the connectors entering in the exit layer will be calculate, the number in the side of neuron are their precipitation

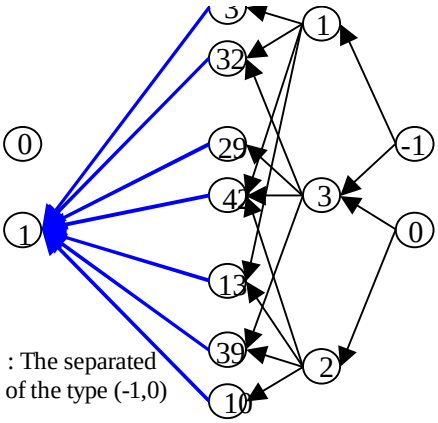


Fig 9.2 : The separated network of the type (-1,0)

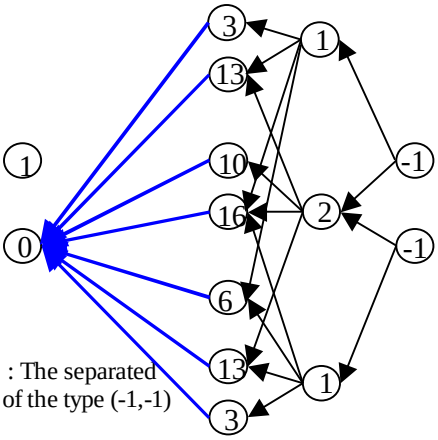


Fig 9.1 : The separated network of the type (-1,-1)

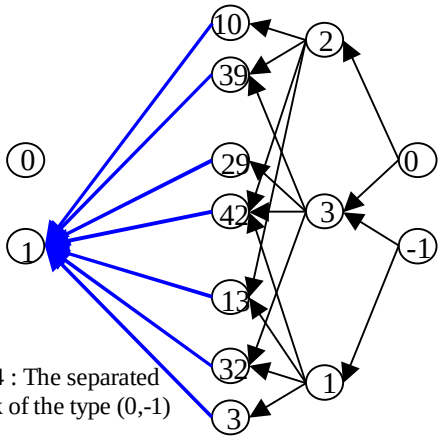


Fig 9.4 : The separated network of the type (0,-1)

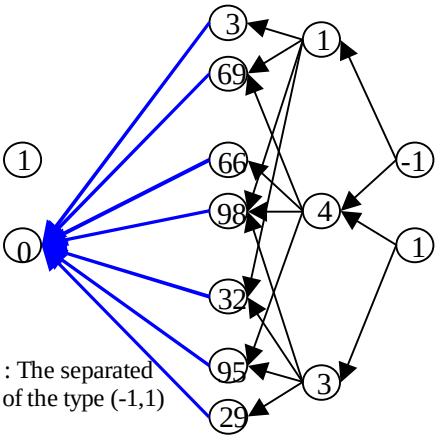


Fig 9.3 : The separated network of the type (-1,1)

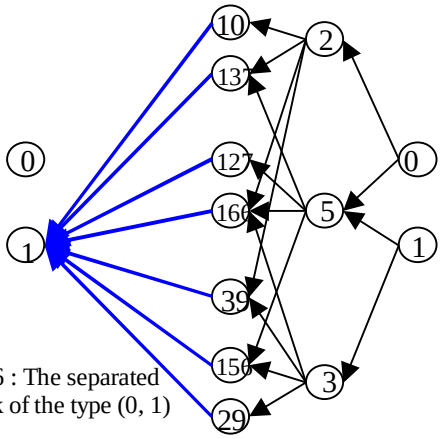


Fig 9.6 : The separated network of the type (0, 1)

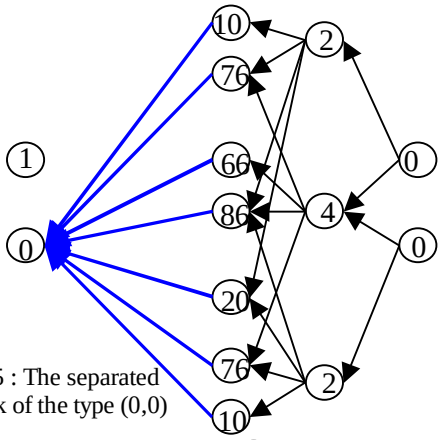


Fig 9.5 : The separated network of the type (0,0)

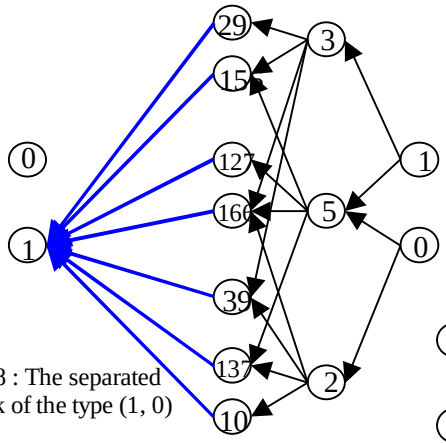


Fig 9.8 : The separated network of the type (1, 0)

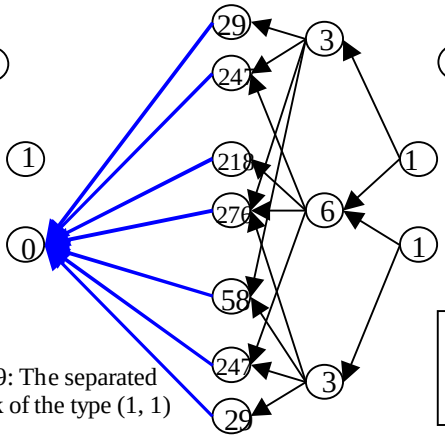
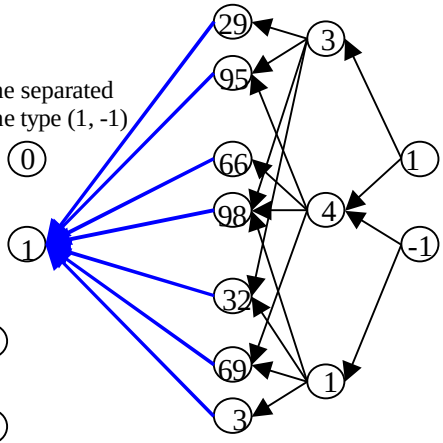


Fig 9.9: The separated network of the type (1, 1)

Fig 9.7 : The separated network of the type (1, -1)



<u>Fig 9</u> : The 9 separated networks of the AXOR problem

1. a+b+29c+2199d+1002e+4098f+218g+2199h+29i=0 ……..(the equation of the (-1,-1) type)
1. a+2b+29c+32770d+24391e+74090f+2199g+59321h+1002i=1 ……..…. (-1,0)
2. a+3b+29c+328511d+287498e+941194f+32770g+857377h+24391i=0……..……(-1,1)
3. 2a+b+1002c+59321d+24391e+74090f+2199g+32770h+29i=1 ……..…..(0,-1)
4. 2a+2b+1002c+438978d+287498e+636058f+8002g+438978h+1002i=0 ……(0,0)
5. 2a+3b+1002c+2571355d+208385e+4574298f+59321g+3796418h+24391i=1……(0,1)
6. 3a+b+24391c+857377d+287498e+941194f+32770g+328511h+29i=0 …….....(1,-1)
7. 3a+2b+24391c+3796418d+2048385e+4574298f+59321g+2571355h+1002i=1 ……....(1,0)
8. 3a+3b+24391c15069225d+10360234e+21024578f+195114g+15069225h+24391i=0... .(1,1)

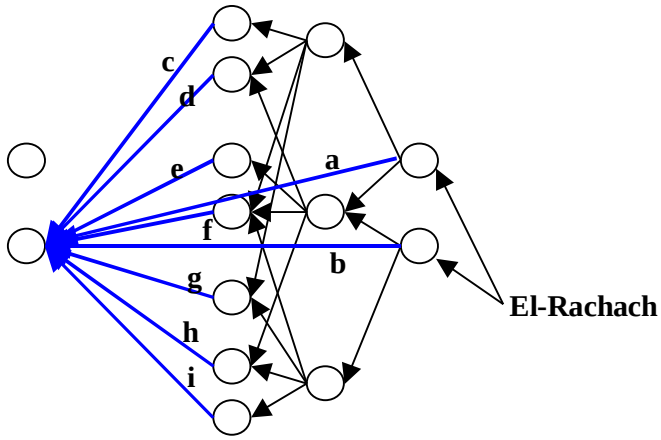


***Fig 10* : The unified network of the 9 separated networks of the AXOR Problem**

a=-0.5443804130451031223184597971981
b=-0.12293981335791926980758946696762
c=-0.0073279235800163311534654550604229
d=4.660931336090446204969308604247 6*10^{-4}
e=-2.8058967125465947513532281660081*10^{-5}
f=-6.6988611821452833961211229781022*10^{-4}
g=0.008848098022236227908791746043 9035
h=3.946595225705397150182586285340 4*10^{-4}
i=-0.0058117948986225388689296317318517

For the first neuron of the exit layer we solve the same equations but we inverse the right members of the equation (we replace 1 by 0 and 0 by 1).

III- Conclusion :

El-dorra neuronal network Technology has proved that the creation of a new hidden layer it's a game of engendering a new no linear variables to complete the linear system of N order, so that the solutions are the weight of the entering connectors of the exit layer. We hope that we present a new importance principle to the world of the neuronal networks.

*"If I am right it is because of my god "**ALLAH**", and if I am wrong it due of my self"*

IV- Thanks

In the first and last to my GOD ***"ALLAH"***
To my father and mother
To all my teachers

V - Reference:

This is an original invention, for this there is not lot of reference

1. [OMA 04] : Omari Abdallah : "Development of a new neuronal networks technology : El-Dorra" paper
2. [OMA&ARA 98] : Omari Abdallah & Arab Naïm : "3D pattern recognition by the neuronal network", engineer thesis from USTO university Oran Algeria