Functional learning & Introduction to the Electronic Brain on
El-dorra neural network technology

OMARI Abdallah
Study & research center of Bioinformatics, Adrar Algeria
omariabdallah@maktoob.com / omariabdellah@yahoo.fr
BP 488 Adrar (RP) 01000, ALGERIA

What is new in this paper?
A different learning method & an electronic brain model

Keywords:
El-Dorra, neural networks, artificial learning, unifying networks, back-forward, nor forward neither back, functional Lagrange interpolation, electronic brain, Separated learning, continued learning, artificial forget.

Abstract:
Habitually in the multi layer perceptron (MLP) learning, we use the back-forward algorithm to change the weights of connectors between neurons but the activate functions stay fixed, the innocent question posed at this moment is: why we don’t make the opposite? So that we can change the activate functions and we fix the weights, the present paper shall discus this solution, also some psychological phenomena as the separated learning, the continued learning and forget, if they can artificially simulated we can construct the first model of the electronic brain.

I.2- Reminder about El-Dorra neural network technology

I.2.1- Presentation

Developed in 2003, this technology applies a psychical phenomena named Separated learning on the ANN, therefore we make a separated network for each class and we use back-forward algorithm to make learning, in the end we unify all this networks in one by adding to each neuron some connectors then we solve a linear system to get there weights, but also there is an other version of this learning methodology named “nor forward neither back” it’s that when we apply the idea only on the exit layer.

More information on: http://groups.yahoo.com/group/el-dorra/
I.2.2- Unifying principle:

If we have two small networks (the based components of El-Dorra net), as below in the next figure, and if we have in the end of the separate learning the weights $w_{11}$, $w_{12}$ for the first network and the weights $w_{21}$, $w_{22}$ for the second network, the two weights of the unified networks $w_1$, $w_2$ can easily counted by solving the linear system:

\[
\begin{align*}
\alpha_1 w_1 + b_1 w_2 &= c_1 \\
\alpha_2 w_1 + b_2 w_2 &= c_2
\end{align*}
\]

\[\begin{align*}
\alpha_1 &= a_1 \\
b_1 &= b_1 \\
\alpha_2 &= a_2 \\
b_2 &= b_2
\end{align*}\]

This principle is applied on all the neurons in El-Dorra net to get the unified weights after making the separated learning. Yet, if we have more than two classes then we need more than two connectors, so we add new connectors from any neurons situated in the previous layers, to get a number of connectors equal to the number of classes and we use the same idea to get the unified network weights eventually.

Another version of this technology is that when we apply the principle of unification directly and only on the exit layer and we fix all the others connectors weight at 1, without using back-forward algorithm, this is what we call "nor forward neither back" version.

I.2.3- Example:

The solution of the XOR problem (nor forward neither back version)

We have two classes, the first represents the "False" class and the second represents the "True" class, nonetheless we have four types: (1,1) and (0,0) in the first one, and (1,0) and (0,1) in the second one, therefore we have four equations, this problem is not linearly separated, thus we must use no linear function as $f(x) = (1+x)^2$ (to eliminate the zero).

- The first neuron in the Exit Layer represents the "False" class and the second neuron represents the "True" class.
- The numbers which are on the top of neurons are their secretions.
- The weights of neurons between the Enter Layer and the Medium Layer=1

Therefore we calculate the entering weights of the exit layer: $X_1$, $Y_1$, $Z_1$, $V_1$ for the first neuron, and $X_2$, $Y_2$, $Z_2$, $V_2$ for the second neuron.
Figure 03: Solution of XOR problem by adding connectors

I.3- Functional learning:
We have seen in the previous that for the unification procedure we use additional connectors and we solve a linear system, but now we use another method to unify networks, it is by changing the activate function; we call this the functional learning

I.3.1- Polynomial Interpolation of function:
In the mathematic, we can get an approximated function for a table of junction values
There are a lot of mathematical methods to develop the approximated function, one of these methods is the Lagrange polynomial interpolation:

\[ P_n(x) = \sum L_{n,i}(x) y_i \]

\[ L_{n,i}(x) = \frac{(x-x_0)(x-x_1)\ldots(x-x_{i-1})(x-x_{i+1})\ldots(x-x_n)}{(x_i-x_0)(x_i-x_1)\ldots(x_i-x_{i-1})(x_i-x_{i+1})\ldots(x_i-x_n)} \]

For the above table we get the Lagrange polynomial: \( P_2(x) = x^2 + x - 2 \) [DER 90]

<Eventually, we get a polynomial degree equal the number of classes -1, and that what gives a good impression of this method. Moreover, the more classes we have, the more our polynomial degree is increased to solve more complicate separation problems.]

In the second version (nor forward neither back), we apply this principle of unification only on the exit layer and we fix all the weight of networks (separated & unified) at the value 1

### I.2.2- Example: The solution of the XOR problem

To eliminate the state of 0 in the table, we can replace 0 by -1 in the desired vector values, and we use the function \( f(x) = (1+x)^2 \)
Figure 05: The solution of the XOR problem by functional learning

So we can deduce the table values of this problem:

For the first neuron of the exit layer:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td></td>
<td>25+81=106</td>
<td>4+9=13</td>
<td>25+36=61</td>
</tr>
<tr>
<td>y=f(x)</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
</tr>
</tbody>
</table>

\[
P_n(x) = \sum_i (L_{n,i}(x) * y_i)
\]

\[
L_{n,0}(x) = (x-13)*(x-61)*(x-40) / (106-13)*(106-61)*(106-40)
\]

\[
L_{n,1}(x) = (x-106)*(x-61)*(x-40) / (13-106)*(13-61)*(13-40)
\]

\[
L_{n,2}(x) = (x-106)*(x-13)*(x-61) / (61-106)*(61-13)*(61-40)
\]

\[
L_{n,3}(x) = (x-106)*(x-13)*(x-40) / (40-106)*(40-13)*(40-61)
\]
So the activate function for the first neuron of the exit layer will be:

\[ F(x) = P_n(x) = \sum L_n(x) y_i \]

\[ = L_{n,0}(x) + L_{n,1}(x) - L_{n,2}(x) - L_{n,3}(x) \]

For the second neuron of the exit layer:

\[
\begin{array}{c|cccc}
I & 0 & 1 & 2 & 3 \\
X & 81+25=106 & 9+4=13 & 36+4=40 & 36+25=61 \\
Y=f(x) & -1 & -1 & 1 & 1 \\
\end{array}
\]

\[ P_n(X) = \sum_i (L_{n,i}(x) * y_i) \]

\[ L_{n,0}(x)=(x-13)*(x-61)*(x-40) / (106-13)*(106-61)*(106-40) \]

\[ L_{n,1}(x)=(x-106)*(x-61)*(x-40) / (13-106)*(13-61)*(13-40) \]

\[ L_{n,2}(x)=(x-106)*(x-13)*(x-61) / (40-106)*(40-13)*(40-61) \]

\[ L_{n,3}(x)=(x-106)*(x-13)*(x-40) / (61-106)*(61-13)*(61-40) \]

So the activate function for the second neuron of the exit layer will be:

\[ F(x) = P_n(X) = \sum_i (L_{n,i}(x) * y_i) \]

\[ = - L_{n,0}(x) - L_{n,1}(x) + L_{n,2}(x) + L_{n,3}(x) \]

**Part II: Introduction to the Electronic Brain**

The most important in the intelligent system is to arrive to the Electronic Brain, in my opinion this model must be able to realise:

**Separated learning:** it can learn each object (class) separately

**Continued learning:** it can learn a new class without returning learning from the beginning with the previous classes.

**Artificial Forget:** in the case of network overflow, it can forget only one object (class) without touching the other classes.

II.1- **Separated learning**:

We have seen this method in the previous mentioned system.

II.2- **Continued learning**:

This is a natural human brain phenomenon, so we can learn new things without starting to learn all objects (classes) from the beginning, while in previous artificial neural network models we must restart the learning of entire classes to integrate a new one.

To add a new class in El-dorra NNT, we need only to integrate the last class term to the Lagrange polynomial

\[ P_{n+1}(x)=\sum [ L_{n,i}(x) * y_i / (x-x_{n+1}) ] + L_{n,n+1}(x) * y_{n+1} \]

\[ = (x-x_{n+1}) * \sum [ L_{n,i}(x) * y_i / (x-x_{n+1}) ] + L_{n,n+1}(x) * y_{n+1} \]

II.3- **Artificial Forget**

The forget phenomena is a positive action to let the brain loses information that is not important then replace it by another information more important in the case of overflowing memory, it's very important to interpret this on an artificial neural network method.

Unlike El-dorra neural technology, the previous ANNs are obliged to erase all the classes and restart the learning from the beginning instead of eliminating only one class.

This phenomenon can be artificially simulated by El-dorra through deleting only the last class term from the Lagrange polynomial, this operation is the opposite of the continued learning.

\[ P_{n+1}(x) = \sum [ L_{n,i}(x) * y_i * (x-x_{n+1}) / (x-x_n) ] - L_{n,n}(x) * y_n \]

\[ = 1/(x-x_n) * \sum [ L_{n,i}(x) * y_i * (x-x_{n+1}) ] - L_{n,n}(x) * y_n \]
Conclusion:

Mathematics and Biology are not sufficient to simulate intelligent systems, the integration of psychology get more important results.

References

[DER 90]: Analyse numérique (numerical analysis) par DERRADGI Salah / University publication office Algeria

Web site:

[Net1] Forum.univbiskra.net/index.php?topic=2396.0;wa p2


[OMA04]: "Development of new neural network technology : El-Dorra", paper accepted by: CICMA'04 Australia, ICBP'04 Sweden, ICIMA'04 China, ICINCO/ANNs'04 Portugal, IKS'04 Turkey, ISNB'04 Holand, Bioinformatics'04 Sweden, MML'04 Italy, SCT'04 USA, CITSA'04 USA, ICDL'04, BICS'04 UK

[OMA05]: "El-Dorra neural network technology, remind & optimisation", paper accepted by: Wesas'05 portugal, NNSC'05 Poland, Isf'05 Texas, ICIT'05 Jordan.

Copyright© 2010 – All right reserved for author®

Thank

In the first & last to my GOD

To my father & mother, to all my teachers,

A special thank for Mr Lagrange to help me to develop Electronic Brain model.