Model of homeostatic artificial neuron

MARTIN RUZEK, TOMAS BRANDEJSKY
Department of informatics and telecommunications
Czech Technical University in Prague
Konviktska 20, Praha 1, 100 00
CZECH REPUBLIC
ruzekmar@fd.cvut.cz, brandejsky@fd.cvut.cz

Abstract: This paper presents a model of homeostatic neuron that is able to find its state of equilibrium by observing the others neurons weights. This method is based on measuring the weights that the other neurons of the neural network assign to the output of the reference neuron, and on improving the parameters of the reference neuron in order to maximize the weights of the other neurons. The basic presumption is that the neuron is trying to maximize its importance in the whole network, which means that it is trying to maximize the values of the weights of the other neurons. The neuron is changing its own input weights and is measuring the reaction of the other neurons. Several types of learning are presented, depending on the way in which the importance of the weights is evaluated.

Key-Words: artificial neuron, homeostasis, learning, artificial neural network

1 Introduction
The design of this model was motivated by intents to simulate brain functions by neural networks. In order to build a neural network it is necessary to define its basic unit, an artificial neuron. Two basic requirements were respected for the construction of this neuron: the similarity to biological neuron (at least in its basic parameters) and the simplicity. The exact copy of biological neuron isn’t achievable and is also not desirable, because we won’t be able to analyze its functions [1].

The basic idea is that the neuron is able to change its weights and by this change to increase its importance in the whole network. In further text, the term input weight \( w_i \) will denote the weight of the connection that is leading the signal into the reference neuron, and the term output weight \( w_o \) will denote the weight of the connection that transfers the output of the reference neuron to others neurons. Utility is real number that quantifies the importance of the reference neuron to other neurons. The utility is calculated as a function of output weights, meanwhile the output weights depend on the the neuron, and therefore on the input weights.

2 Methodology
The proposed neuron is based on McCulloch-Pitts model [2] of artificial neuron that is illustrated on picture 1.

Mathematically, its function is described as \( f(x) = (\sum_{i=1}^{n} x_i w_i + \delta) \). In this paper, a new improvement of this model is presented. The proposed neuron is able to improve automatically its function in a manner that is similar to biological neuron. The similarity to biology was one of the basic requirements; therefore the back propagation algorithm [3,4] can’t be used. The reason is that in this model the backward connections are identical to forward connections. There are no special communication channels for the back propagation of the error. The neuron doesn’t calculate the difference between desired and real value, it only observes the weights of the output connections. The basic presumption is that the neuron seeks to be as useful as possible, which means that it intends to make the other neurons to set their input weights to the greatest possible value. The weights are limited to interval \([-1,1]\) because of practical reasons. With the respect to the negative values, it isn’t
possible to determine the importance of the connection directly by its weight. For instance, a connection with weight \(-0.8\) is more important than 0.3. Therefore, the absolute values or the squares of the weights are used instead of the values directly.

The process of the learning of the neuron can be described by the following algorithm:

1. neuron sets its input weights to random values
2. neuron performs the forward phase (the computation of the output values)
3. neuron evaluates the output weights
4. neuron adds \(dw\) to the weight of the first connection
5. neuron repeats the forward phase; computes the output of the same input with the changed weight
6. if the previous change made the output weights greater, the neuron will confirm the change. In the contrary, it will take off \(dw\) from the first connection
7. neuron repeats steps 3 to 6 with all connections and all inputs

This algorithm can be used for neural networks with one layer. In networks with more layers, there will be different delays of the signal with the information about the utility. These delays will cause instabilities that will make impossible the direct use of this method because in these multilayered systems the change of the input doesn’t influence the output immediately. For neural networks with more layers the algorithm must be modified. This work will deal only the learning of a neuron as a part of one layered neural network.

To calculate the importance of the neuron, several methods based on different theories can be used. The first extreme case is searching such setting of the input weights that maximizes the sum of absolute values of the output weights. In other words, the neuron is trying to be interesting to all of the neurons in the higher layer. The other extreme case is neuron that is trying to be interesting for only one neuron in the higher layer. This neuron is searching a setting for which the maximum of the output weights is maximal. Between these extremes there are many compromise variants. For example, the neuron may try to increase its importance to some given number of output neurons.

During the process of learning, the neuron can set the weights, the slopes and the thresholds. The process of setting of the slopes and the thresholds is analogical to the weights. In this article, only the weights adjustment will be discussed.

### 2.1 Learning of the neuron by the sum of the output weights

This method is finding such vector of input weights \(\Lambda = \{w_1, w_2, \ldots, w_n\}\) for which the sum of the absolute values of the output weights is maximal. This model corresponds well to the biological neuron, because there is only one axon leaving the biological neuron and therefore the neuron can only know the total amount of the signal that is accepted by the others, not the particular weights. In the case of artificial neuron, we expect also the negative weights. Because of that, the neuron will sum the absolute values. The utility \(q\) is:

\[
q = \sum_{j=1}^{n} |w_j^o| \quad (1)
\]

Another option is to use the square values:

\[
q = \sum_{j=1}^{n} (w_j^o)^2 \quad (2)
\]

Method according to (2) will lead to different values than (1), as it prefers the changes of high absolute values. The neuron based on (2) won’t be interested in improving the output weights that are close to 0, as the improvement of these weights will have smaller impact on its overall utility. For example, the combination of the output weights \([0.5, 0.6]\) is better than \([0.1]\) when considered according to (1), otherwise the second combination is better.

The neuron must be equipped with a memory and a function that enables the computation of its utility. The memory makes possible the comparison of the current utility with at least one of the past values. The neuron doesn’t know the number of the outputs and therefore doesn’t know the maximum of the utility \(q\). Therefore it will never be aware of reaching the optimal homeostatic position. The process of the learning will last for the whole time of the existence of the neuron.

### 2.2 Learning of the neuron by the maximum of the output weights

Another possible type of training is based on the idea that the neuron is willing to increase its importance to only one neuron in the output layer. In this case the optimal setting maximizes
function \( q = \max |w_j'|; \ j = \{0, 1, \ldots, n\} \). In this situation, the neuron can see whether its setting is ideal or not by comparing its utility \( q \) to maximum value, that is 1. When \( \max |w_j| = 1 \), no further improvement is possible. The problem is that in situation with many output neurons, the probability that at least one of them is close to 1 is high. Then there will be no learning since the initialization of the process because the neuron will be close to its optimal position. This problem can be solved by adding another criterion that will take into account more neurons. There are many other ‘compromise’ solutions that use advantages of both methods. One example is a neuron that is trying to maximize its importance to some given number of neurons in the output layer.

### 3 Results

The neuron was programmed in MATLAB R2008b. The function of learning is based on methods described in Part 2. The input is a vector of any length composed by ones and zeros. The output is a real number from (0, 1) interval. This number determines the values of the output weights, which are the ‘second input’ of the neuron. The external parameter sets which function will be used for evaluation of the quality of the setting (sum of absolute values, square, maximum or other).

An important step in the design phase of this model was the definition of the output layer. This layer is necessary because the previously described neuron can be tested and improved only as a part of functional virtual environment. The model of the output layer simulates the vector of the connections between the reference neuron and the higher layer. The realization of this layer was the major difficulty of the whole model. The neuron’s functions are well defined and therefore its code can exactly fulfill its functions, but the output layer has many dubiousness and ambiguities.

In the output layer there may be many different neurons with diverse functions. There is no practical limitation of the number neither of the output neurons nor of their functions. During the first phase of the testing, simple and homogenous layers were considered. Neurons of the output layer were more or less identical. The great majority of the neurons was interested in only one input of the reference neuron. In this case, the reference neuron tends to set one weight (the desired connection) to 1 and all the others to 0.

This means that there is one dendrite with useful signal and all the other dendrites transmit useless noise (from the point of view of the higher layer). The speed of convergence for different number of dendrites is in Table 1. In this experiment it was assumed that the number of the inputs is equal to the number of the output units. Especially in the experiments with few dendrites the random character of the initial setting has a great influence on the number of iterations. To limit the importance of the random initial setting, the final result was calculated as an average of ten experiments with the same conditions.

<table>
<thead>
<tr>
<th>Number of dendrites</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of iterations</td>
<td>137</td>
<td>274</td>
<td>698</td>
<td>2825</td>
</tr>
<tr>
<td>Number of dendrites</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>Number of iterations</td>
<td>4891</td>
<td>2818</td>
<td>3714</td>
<td>4528</td>
</tr>
</tbody>
</table>

Table 1: Number of necessary iteration as function of number of the inputs to the neuron

In the next step, more complex functions were desired by the output layer. First, the neurons were divided into two groups, each of which was interested in another input signal of the reference neuron. In this case, the convergence process was significantly slower. In this test, the neuron is not considered to be trained immediately after reaching the desired level of importance, but must be able to fulfill the desired function for some period of time. In the following phase, the neuron was tested in an environment with diverse and complex desired functions. The output layer was interested in a group of functions that can’t be realized simultaneously at the same time. In this case, the convergence was very slow and sometimes the neuron didn’t read the homeostatic position at all.

### 3.1 Discussion

The main advantage of the proposed neuron is its ability of self-learning in way that can be expected in the biological neuron. The learning is indirect; there is no channel for the back propagation of the error. There is also no external function that describes the desired work of the neuron. Instead of this, the neuron is approving itself in order to increase its importance to other neurons. This fact implies that it can be trained incorrectly. If the other neurons are interested in
incorrect data, the neuron will try to provide them. The learning is slower than with backpropagation algorithm [5].

The basic disadvantage of this type of learning is the delay. The proposed neuron changes its weights and expects that the change will be immediately reflected in the output layer. This presumption will be true only for two layered networks. In the case of multi-layered network it will take several steps until the change appear again in the neuron. One of the possible solutions of this problem is setting the dynamics of the inputs to enough low level, so that the change of the input signal will be significantly slower than the communication between the neurons. [6]

4 Conclusions

The neuron proposed in this article is able of independent learning. During this process it is searching its homeostatic position. The equilibrium state of the neuron is a situation when the acceptance of its out is maximal. The neuron is trying to increase its significance by changing the input weights. This process can be adapted also for other parameters of the neuron, such as the slope and the threshold. The utility of the neuron is measured by the weights that other neurons use to multiply the output of the reference neuron. The main advantage of this method is that the process of learning is autonomous, no external learning function is necessary. The experiments confirmed that this neuron converges to the homeostatic position. The simpler the desired function is, the faster is the convergence. In case of difficult and diverse desired function the neuron doesn’t converge. The main disadvantage of this model is that is applicable to systems with first order delay. In the following research this model will be adapted to networks with more layers. This model may be a foundation stone of a new kind of neural network. The network composed of independent homeostatic neurons may be used for the simulation of brain functions, as this neuron was inspired by biological neuron. The use of fuzzy models seems to be perspective in this area [7]. However, the possible use of this network may be much wider, as it can be used in many different areas and tasks. In future research, we will focus on adapting this neuron for the whole network.

References

[5] P. Radonja, S. Stankovic, D. Drazic, B. Matovic, Generalized Models Based on Neural networks and Multiple Linear Regression, Proceedings of the 5th WSEAS Int. Conf. on CIRCUITS, SYSTEMS, ELECTRONICS, CONTROL & SIGNAL PROCESSING, Dallas, USA, November 1-3, 2006 279-284