Optimum size of feed forward neural network for Iris data set

Wojciech Masarczyk
Faculty of Applied Mathematics
Silesian University of Technology
Gliwice, Poland
Email: wojcmas042@polsl.pl

***Abstract*—This paper illustrates a process of finding optimum structure of neural network which will be used to solve problem of Iris data set. This work presents dependencies between number of layers, neurons and efficiency of network providing the best configuration for given data set.**

***Index Terms*—feed forward neural network, Iris data set, backpropagation algorithm**

I. Introduction

Computers can help on automated models implemented to co-work with other devices in service. We can find various application of intelligent systems in medicine, technology, transport, etc. We expect from computer to control possible dangers and advise the best options, while assisting humans [3], [4]. However in these aspects it is necessary to process data of various origins [13]. We have many possible approaches to data analysis. There are solutions devoted to big data [8], where we use some sophisticated methods implemented for knowledge engineering.

Mainly computational intelligence is assisting in data processing to discover knowledge from input data, there are mathematical models implemented to find incomplete information and work with this issues [1], [2]. Similarly we can find reports on efficiency of neural networks in processing incoming information [10]. Neural networks are efficient in processing input data of various types, from voice samples [15] to handwritings [11]. However to improve efficiency of processing it is important to discuss optimum size of these architectures, what will be done in this article using iris data set as an example.

II. Neural network model

A. Biological inspiration

Neural network is a biologically inspired model of mathematical computations which structure is based on architecture of human brain. In the simplest approach brain consists of approximately $8.6 \cdot 10^{10}$[5] neurons that are connected with each other creating a network of 10^{15} connections through which impulses are being sent simultaneously resulting in brain processing speed close to 10^{18} operations per second.

Why one should even bother applying neural network into any field of computations? Mostly due to the fact that artificial neural networks are able to generalize obtained knowledge which means that after proper training such network should be able to predict correctly value of given example not included in training data. Second advantage of these models is robustness for random fluctuations or lack of part of values in data set. Generally neural networks are used to solve problems that seems to be incomputable or too complicated to solve by classical algorithms.

B. Artificial neuron

One can observe that artificial neuron is really simplified versions of human neuron, however still keeps three most important features of neuron:
-taking input from other neurons (dendrites)
-computing and processing impulses taken as input
-sending on computed impulse to further neurons(axon)

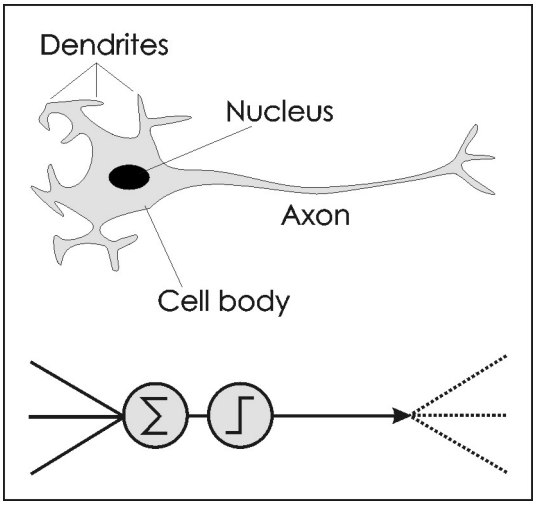

Figure 1. Biological and artificial neuron

One cycle of computations in neurons may be described as follow: values are being multiplied by appropriate weights and summed together in a cell of neuron after which sum is taken as argument to activation function which value is set to be impulse send as the output of neuron. Equation describing this process is presented below:

$$y = f(\sum_{i=1}^{n} w_{(x_i)j} \cdot x_i), \tag{1}$$

where
$w_{(x_i)j}$ – weight between i and j neuron in consecutive layers,
x_i – value of neuron i,
f – activation function

In (1) it is assumed that previous layers consists of n neurons.

C. Activation function

Activation functions is an abstract indicator that says about action taken in neuron. In simplest case once information is important it outputs 1 otherwise neuron outputs 0. This model is called a binary step function, despite its simplicity it is able to solve a few problems, however it is insufficient with more complex problems due to the lack of desired features detailed in following list:
- Finite range results in stability while gradient - based method are used for learning
- Continuous differentiable is necessary for every gradient - based method, because of that feature binary step function cannot be used in models with gradient based learning algorithms.
- Identity near origin fasten the learning process once initial weights are small numbers.
In this paper results will be obtained using only four different activation functions:

Logistic function:

$$f(x) = \frac{1}{1+e^{-\beta x}} \tag{2}$$

where
β – determines steepness of function near origin,

Hyperbolic tangent:

$$f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1 \tag{3}$$

Arctangent

$$f(x) = \frac{1}{\tan x} \tag{4}$$

Modified Hyperbolic tangent:
Modification proposed by Yann LeCun [6]

$$f(x) = 1.7159 \tanh(\tfrac{2}{3}x) \tag{5}$$

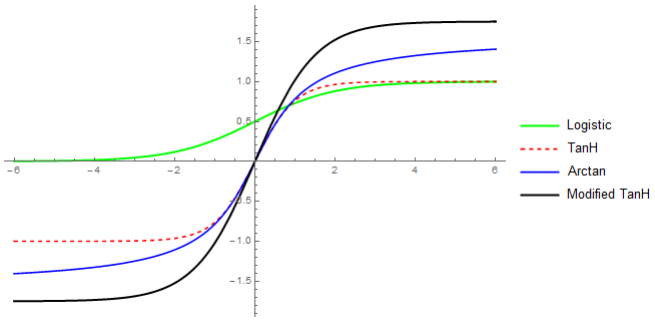

Figure 2. Graph of different activation functions tested in this paper

D. Neural network

Feed forward neural network is constructed by neurons stacked in a rows called layers that are fully connected with previous and subsequent layer. First and last layer are respectively called input and output layer, each layer between these two is called a hidden layer. Role of neurons in input layer is to store the initial data and send it further. Flow of information takes place from left to right, so once initial data is provided the result will appear at the output layer which ends one full cycle of computations for network.

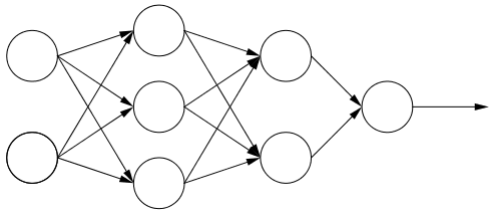

Figure 3. Example of feed forward neural network with 2 hidden layers

According to (1) output of neural network is just a sum of random weights multiplied by initial values, the only parameter that might be modified in neural network are weights so learning algorithm is just a process of changing weights in such way that output is exactly the same as presented in data set. In this paper backpropagation algorithm will be discussed.

E. Backpropagation algorithm

To train the network it is necessary to provide a data set that consists of vector of input signals $(x_1, x_2, ..., x_n)$ and corresponding desired output z. Thanks to this fact it is possible to compute the difference between output signal y and desired value z. Let $\delta = z - y$. Next step is to propagate δ error through every neuron according to equation:

$$\delta_j = \sum_{i=1}^{n} w_{(j)i}\delta_i, \tag{6}$$

where
$w_{(i)j}$ – weight between i and j neuron in consecutive layer,
δ_i – value of error on neuron i in consecutive layer,

Once δ error is computed for each neuron following step is to actualize weights using equation below:

$$w_{(j)i} = w_{(j)i} + \eta\delta_i \frac{df_i(x)}{dx} y_j \tag{7}$$

where
$w_{(i)j}$ – weight between i and j neuron in consecutive layers,
y_j – value of neuron j,
f_i – activation function on neuron i
δ_i – value of error on neuron i in consecutive layer,
η – coefficient that affects speed of learning.

Learning is an iterative process that takes place until δ for output layer is smaller than desired precision.

III. Convergence of gradient descent algorithm with respect to η

While analyzing equation (7) it is obvious that η has a direct impact on pace of learning of neural network since derivative determines direction where error function descends. It is crucial to select balanced value of η. Too small value may result in slow learning process while value bigger than necessary will end with divergence of algorithm. Four different scenarios of η values are presented in graphical interpretation. Every test in this paper was carried out with $\eta = 0.02$ as it appears to be most suitable value of this parameter.

IV. Iris data set

Iris data set is a classic and one of the best known sets for pattern recognition described in [7]. It consists of 150 instances that are equally divided into three different classes. Each class refers to a different type of iris plant. Every sample is a vector of length equal four, where vector components describes:
1. sepal length [cm]
2. sepal width [cm]
3. petal length [cm]
4. petal width [cm].

In order to create training data set, 35 samples were chosen from each class, so that 30% of whole set is left for testing accuracy of trained model. The aim for neural network is to classify Iris, described by these four dimensions, into one of three given classes.

V. Tests

For further tests η is equal to 0.02, it will be pointed out once η will change. In every case desired error is set to be smaller than 10^{-5}. Maximum epochs are set to 20000.
In order to compare obtained results it is necessary to come up with general formula determining number of weights in single network:

$$q = 4 \cdot 3 \prod_{i=1}^{n} i, \tag{8}$$

where
i – neurons on each hidden layer,
n – number of hidden layers,

As batch learning approach is used in this case every iteration is equal to updating every weight in whole neural network structure. Since then it seems natural to come up with equation describing numbers of updates done throughout whole process of learning each network.

$$Q = q \cdot r \tag{9}$$

where
r – number of iterations,
q – number of neurons in network,

Q from equation (9) will be used in next section to compare efficiency of different network structures.

A. Results

It is clearly visible in table below that in case network consists of only input and output layers (0 hidden layers) it fails to classify properly each of three types, usually being able to classify correctly two of them while having issues with third one. This is caused by the fact that one class is linearly separable from the other two, however the latter are not linearly separable from each other.

One can observe that Hyperbolic Tangent function is most uniform function while being most effective taking into consideration number of iterations and number of neurons needed to obtain desired precision. Similar behaviour can be viewed with Arctangent function, however it fails to work for networks that were properly computed with Hyperbolic Tan.
On the other side Modified Tangent performed below expectations, however it is important to remember that all tests were carried out for fixed η. Since steepness of this function is modified and value of η should be set with respect to derivatives [6] its underperformance seems to be understandable.

After analyzing table it is clear that adding additional hidden layers is not worth the effort since optimum is obtained for 1 or 2 hidden layers. Note that each hidden layer with 10 neurons added have to result in decreasing iterations tenfold to be computationally profitable, unfortunately this does not happen with any function. Moreover, as shown, network that is too big (5 hidden layers and more) tend to fail this classification task at all. It might caused by the fact that Iris Data set is relatively small (150 examples) and it is not enough to adjust properly so many weights.

Three graphs presented in this paper represents Error function over consecutive iterations. Figure 6 presents nearly perfect plot that is smooth and monotonic at most of a time. On contrary to the Figure 5 which presents heavy oscillations of Error function caused by wrong η coefficient. Exactly the same situations appears to be on Figure 7 at around 400 iteration, luckily it is a situation when $\eta > \eta_{opt}$ and after a few iterations of oscillations it finally converges. Values presented in table show that network failed to obtain desired precision in 20000 iterations.

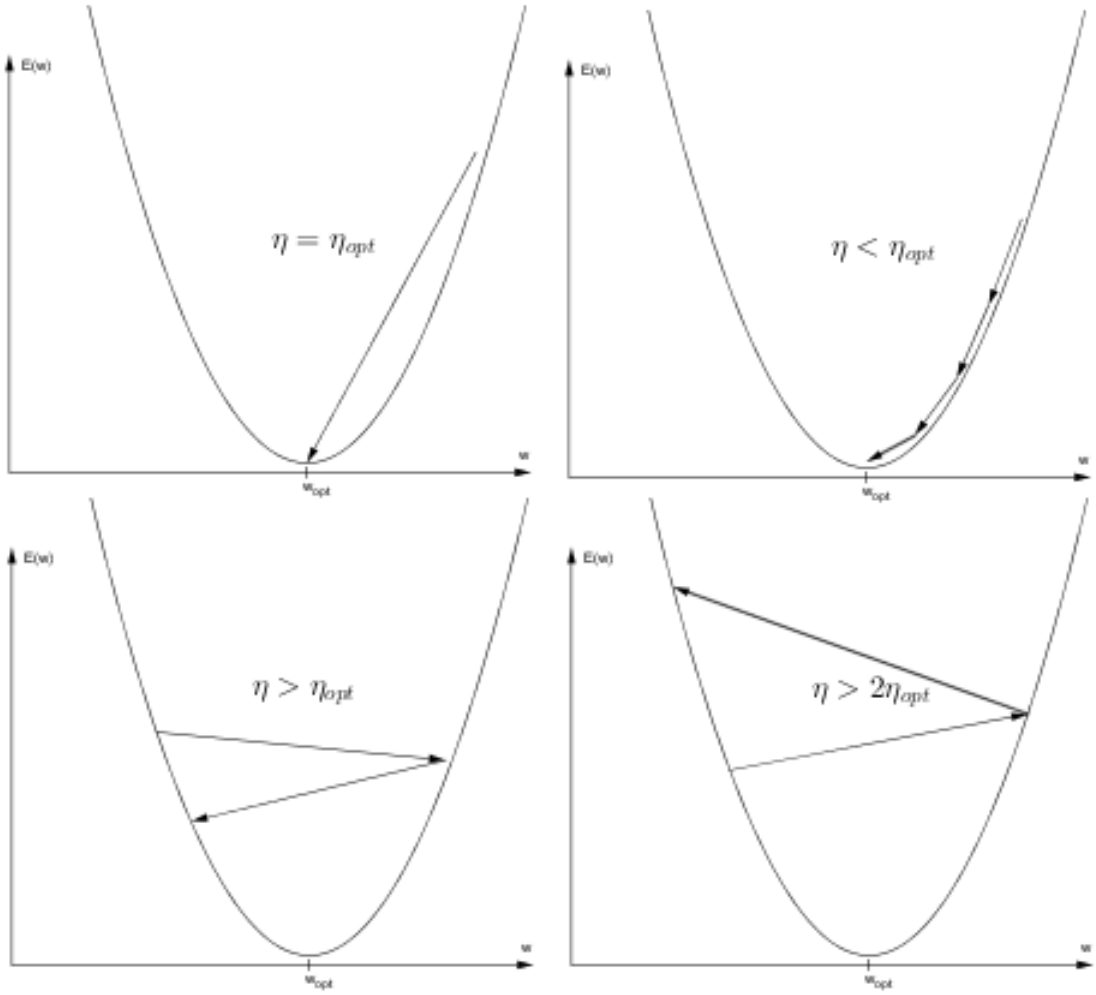

Figure 4. Graphical interpretation of η coefficient.

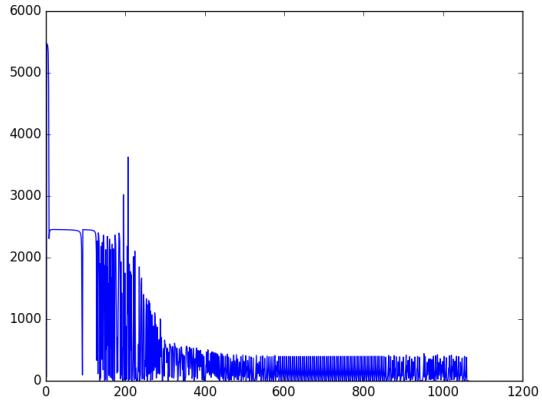

Figure 5. Error function over iterations in 0 hidden layer network

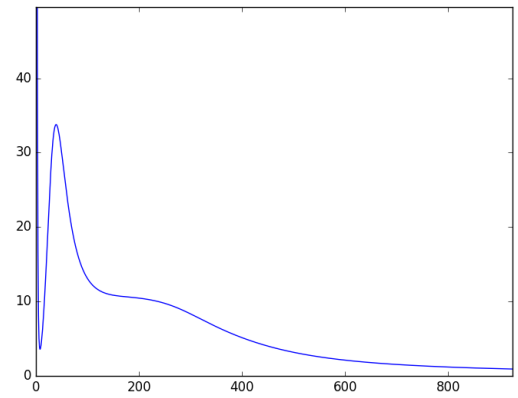

Figure 6. Error function over iterations in 1 hidden layer network

VI. Remarks

During designing artificial neural network there is a list of crucial parameters that need to be set precisely in order

Activation function	Number of hidden layers	Neurons on hidden layers	Total weights	Iterations	Weights · iterations
Logistic Function	0	0	12	F	F
	1	5	60	F	F
		10	120	10811	1297320
	2	10 - 10	1200	8467	10160400
		10 - 5	900	F	F
		5 - 10	900	5909	5318100
	3	8 - 8 - 8	6144	F	F
		10 - 8 - 6	10800	3867	41763600
		6 - 8 - 10	10800	553	5972400
	4	6 - 6 - 6 - 6	15552	F	F
		10 - 9 - 8 - 6	51840	F	F
		6 - 8 - 9 - 10	51840	F	F
	5	7 - 7 - 7 - 7 - 7	201684	F	F
		12 - 10 - 9 - 8 - 7	725760	F	F
		7 - 8 - 9 - 10 - 12	725760	F	F
Hyperbolic Tangent	0	0	12	F	F
	1	5	60	259	15540
		10	120	2047	245640
	2	10 - 10	1200	1124	1348800
		10 - 5	900	1266	1139400
		5 - 10	900	F	F
	3	8 - 8 - 8	6144	F	F
		10 - 8 - 6	10800	313	3380400
		6 - 8 - 10	10800	1359	14677200
	4	6 - 6 - 6 - 6	15552	1512	23514624
		10 - 9 - 8 - 6	51840	1566	81181440
		6 - 8 - 9 - 10	51840	874	45308160
	5	7 - 7 - 7 - 7 - 7	201684	1072	216205248
		12 - 10 - 9 - 8 - 7	725760	F	F
		7 - 8 - 9 - 10 - 12	725760	F	F
Arctangent	0	0	12	F	F
	1	5	60	699	41940
		10	120	311	37320
	2	10 - 10	1200	2824	3388800
		10 - 5	900	2567	2310300
		5 - 10	900	4026	3623400
	3	8 - 8 - 8	6144	F	F
		10 - 8 - 6	10800	777	8391600
		6 - 8 - 10	10800	831	8974800
	4	6 - 6 - 6 - 6	15552	1072	16671744
		10 - 9 - 8 - 6	51840	235	12182400
		6 - 8 - 9 - 10	51840	F	F
	5	7 - 7 - 7 - 7 - 7	201684	F	F
		12 - 10 - 9 - 8 - 7	725760	F	F
		7 - 8 - 9 - 10 - 12	725760	F	F
Modified Tangent	0	0	12	F	F
	1	5	60	248	14880
		10	120	1245	1297320
	2	10 - 10	1200	F	F
		10 - 5	900	F	F
		5 - 10	900	F	F
	3	8 - 8 - 8	6144	F	F
		10 - 8 - 6	10800	F	F
		6 - 8 - 10	10800	F	F
	4	6 - 6 - 6 - 6	15552	F	F
		10 - 9 - 8 - 6	51840	F	F
		6 - 8 - 9 - 10	51840	F	F
	5	7 - 7 - 7 - 7 - 7	201684	F	F
		12 - 10 - 9 - 8 - 7	725760	F	F
		7 - 8 - 9 - 10 - 12	725760	F	F

to achieve properly working model. As most important and discussed in this paper are:

- Activation function - as turned out, it has a huge impact on capabilities of network, not only these provided by differentiability but also can significantly accelerate process of learning,
- Learning coefficient - highly depends on activation function, is a crucial factor in converging process of gradient - based methods of learning,
- Number of hidden layers and neurons - in simple problems like that one discussed in this paper 2 hidden layers are enough,

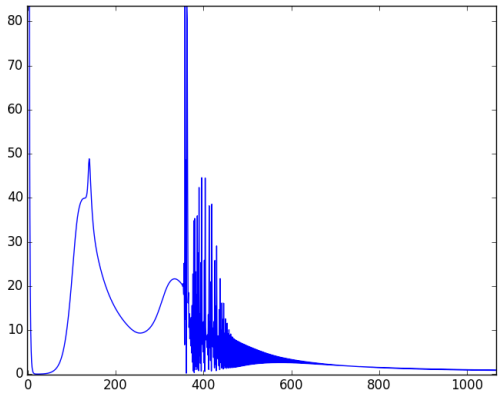

Figure 7. Error function over iterations in 2 hidden layer network

- Number of iterations and desired precision - again, strongly depends on problem one tries to solve, there is no general rule that describes how to set these values, however it is important to set upper limit of iterations in order to stop the computations at some point once learning algorithm is unable to achieve desired precision,

VII. Summary

Among various architectures neural networks are one of most efficient processors of information about controlled objects, moreover these architectures can cooperate with information presented in a form of image, voice sample, network statistics, etc.

Each implemented solution needs adjusted architecture to exactly fit the model of decision support. Therefore research on performance of various types can give valuable information about performance, what can improve implementations of neural networks.

References

[1] P. Artiemjew, Stability of Optimal Parameters for Classifier Based on Simple Granules of Knowledge, *Technical Sciences*, vol. 14, no. 1, pp. 57-69. UWM Publisher, Olsztyn 2011.

[2] P. Artiemjew, P. Gorecki, K. Sopyła, Categorization of Similar Objects Using Bag of Visual Words and k – Nearest Neighbour Classifier, *Technical Sciences*, vol. 15, no.2, pp. 293-305, UWM Publisher, Olsztyn 2012.

[3] R. Damasevicius, M. Vasiljevas, J. Salkevicius, M. Woźniak, "Human Activity Recognition in AAL Environments Using Random Projections" *Comp. Math. Methods in Medicine*, vol. 2016, pp. 4073584:1–4073584:17, 2016, DOI: 10.1155/2016/4073584.

[4] R. Damasevicius, R. Maskeliunas, A. Venckauskas, M. Woźniak, "Smartphone User Identity Verification Using Gait Characteristics" *Symmetry*, vol. 8, no. 10, pp. 100:1–100:20, 2016, DOI: 10.3390/sym8100100.

[5] S. Herculano-Houzel, The human brain in numbers: a linearly scaled-up primate brain, *Frontiers in Human Neuroscience* vol. 3, pp. 31:1-31:11, 2009, DOI: 10.3389/neuro.09.031.2009

[6] Y. Le Cun, L. Bottou, G. Orr, K. Muller, Efficient backprop. Neural Networks: Tricks of the trade, 1998.

[7] R. Fisher, The use of multiple measurements in taxonomic problems, *Annual Eugenics*, vol. 7, pp. 179-188, 1936.

[8] Z. Marszałek, "Novel Recursive Fast Sort Algorithm," in *Communications in Computer and Information Science*, vol. 639, pp. 344–355, 2016, DOI: 10.1007/978-3-319-46254-7_27.

[9] C. Napoli, E. Tramontana, E., "An object-oriented neural network toolbox based on design patterns," in *International Conference on Information and Software Technologies*, pp. 388-399, 2015, DOI: 10.1007/978-3-319-24770-0_34.

[10] C. Napoli, G. Pappalardo, E. Tramontana, "A mathematical model for file fragment diffusion and a neural predictor to manage priority queues over BitTorrent," in *Applied Mathematics and Computer Science*, vol. 26, no. 1, pp. 147-160, 2016, DOI: 10.1515/amcs-2016-0010.

[11] D. Połap, M. Woźniak, "Flexible Neural Network Architecture for Handwritten Signatures Recognition" *International Journal of Electronics and Telecommunications*, vol. 62, no. 2, pp. 197–202, 2016, DOI: 10.1515/eletel-2016-0027.

[12] C. Napoli, G. Pappalardo, G. M. Tina, E. Tramontana, "Cooperative strategy for optimal management of smart grids by wavelet rnns and cloud computing" *IEEE transactions on neural networks and learning systems*, vol. 27, no. 6, pp. 1672–1685, 2016, DOI: 10.1109/TNNLS.2015.2480709.

[13] D. Połap, M. Woźniak, "Introduction to the Model of the Active Assistance System for Elder and Disabled People," in *Communications in Computer and Information Science*, vol. 639, pp. 392–403, 2016, DOI: 10.1007/978-3-319-46254-7_31.

[14] C. Napoli, G. Pappalardo, E. Tramontana, R. K. Nowicki, J. T. Starczewski, J. T., M. Woźniak, "Toward work groups classification based on probabilistic neural network approach" in *International Conference on Artificial Intelligence and Soft Computing*, pp. 79–89. 2016, DOI: 10.1007/978-3-319-19324-3_8.

[15] D. Połap, "Neuro-heuristic voice recognition," in *2016 Federated Conference on Computer Science and Information Systems, FedCSIS 2016, Proceedings*. 11-14 September, Gdańsk, Poland, IEEE 2016, pp. 487–490, DOI: 10.15439/2016F128.