Identification and Clusterization of Images with Neuron Networks

Vladimir A. Yanakiev[1] and Milena K. Lazarova[1]

Abstract – **Neuron networks NN are computer models, which can "learn by their expirience". They are based on human brain on some ways. In this research is made detailed comparison between Self Organizing Map (SOM) and Feed - forward neural network (FFNN). Because of this neurons of input layer are same number with same meaning. Used input data are taken after computing from binary images of fencing weapons. Input data can be output of another application (for generation, graphical computing or even another NN). Different possibilities for outputs in Both NN are examined. FFNN without hidden layer is investigated in application.**

Keywords – **Neural network, FFNN, SOM, backpropagation, Image identification, clusterization.**

I. Introduction

One of the first comparisons in this sphere is made by BERNARD WIDROW [3], without detailed analog. This research is based on description between different NN and accent is theoretical.

In following presentations [4-6] image identification with FFNN is study. By our knowledge in these researches investigations doesn't seems to be reported the usage of FFNN without middle layer. Reyes-Aldasor is made image identification with SOM [7]. This research is only for one class detection.

Bajwa [8] use different kind of NN for image classification, use component analysis model and compare results, but without analyze of the experiments and NN.

All of the authors use segmentation of the image and inputs of NN are the average values of these segments. In this research is used another way, in which characteristic indications like inputs are used. Characteristic indications are availability of given topology, distance between points or relations between distances. This data saved on this way is with small size.

II. Experimental

Short Algorithm description:

FFNN is one of the most popular NN. NN of this type have one input layer, output layer, and may have hidden layers. Neurons of every layer are connected with others by weight. Information waves are from input to output. Hidden layer is changing linear of NN. When output signals are linear, according to the input signals it is possible to avoid hidden layers, according Gochev [1] in this case they repeat inputs in some form, but it's could be applicable in many cases. The role of every neuron is to compute input signals. And output of neuron is sum of input multiplied by signals from previous layer (1).

$$N_{etj} = \Sigma W_{j,k} * Ou_{tk} \tag{1}$$

On the base of difference between target output vector and calculated output vectors are calculated errors. Errors are propagating in previous layers and change weights.

NN with hidden layer is shown on Fig.1a. NN without hidden layers is shown Fig.1b.

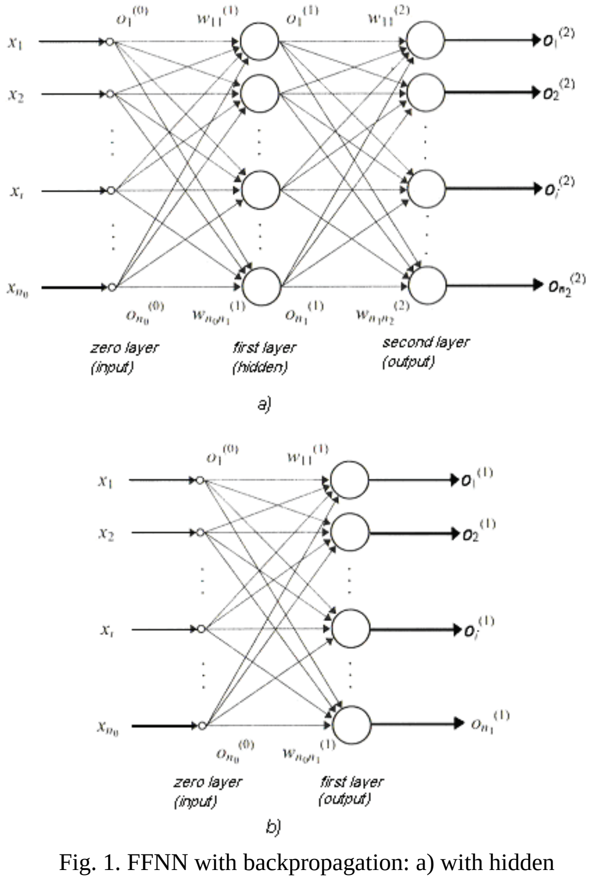


Fig. 1. FFNN with backpropagation: a) with hidden layers and b) without hidden layers.

One pass of all input vectors is calling epoch and many of epochs are enough for entire learning of NN. After this NN is

[1]Vladimir A. Yanakiev is with the Institute of Computer an Communication System, Bulgarian Academy of Science, E-mail: vladi_ian@yahoo.com

[2]Milena K. Lazarova is with the Faculty of Computer Sciences and Technology, Technical University of Sofia, Bulgaria, E-mail: milaz@tu-sofia.bg

able predictable to process data, which is not included in training vectors. This is ability to compute of unknown objects, predictable ability.

Kohonen suggest NN with algorithm without teacher, Self organizing Map (SOM) with clusterization algorithm [2]. One layer NN. Its element formed 2 Dimension matrixes. Refer to Fig. 2.

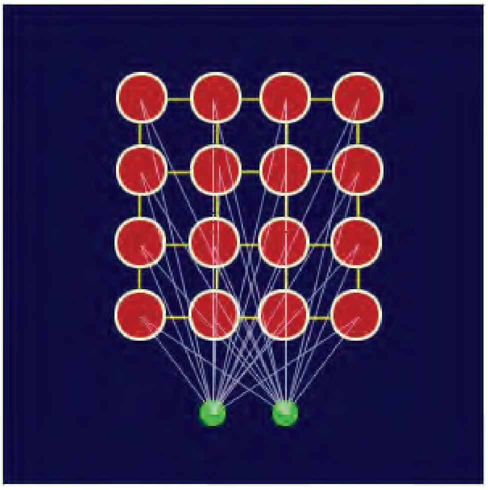


Fig. 2. Sample NN of Kohonen

The Input is usual one dimension vector with N elements. Every Input Element is connected with every output of the matrix. Input values are presented in time without target. Clusterization in NN is forming correlated input vector with weights of connections. After enough input set cluster center are formed. There are two phases: Comparing input vector and modification of weights.

Training occurs in a lot of iterations. Steps of the algorithm:

- Each node's weights are initialized.
- A vector is chosen at random from the set of training data and presented to the lattice.
- Every node is examined to calculate which one's weights are most like the input vector. The winning node is commonly known as the Best Matching Unit (BMU).
- The radius of the neighbourhood of the BMU is now calculated. This is a value that starts large, typically set to the 'radius' of the lattice, but diminishes each time-step. Any nodes found within this radius are deemed to be inside the BMU's neighbourhood.
- Each neighbouring node's (the nodes found in step 4) weights are adjusted to make them more like the input vector. The closer a node is to the BMU, the more its weights get altered.
- Repeat step 2 for N iterations

Characteristics of SOM and FFNN

Input data:

In image identification is possible to use some different kinds of inputs here are some of them. All inputs are applicable for FFNN and SOM:

- Segments of image: image is devided in segments and one every segment is found average value of colour or direction (vector) or combination. Disadvantage is necessary of similar location of image.
- Pixels of image: this is an isolated case of previous, every pixel is an input and input neurons are as many as pixels in image. This network is not applicable in big images. In NN like this even one pixel offset would make application useless. It’s applicable for example for letters from same font. The advantage is the missing of graphical computing.
- Characteristic indications: In this case specific topology, distance specific points or relations between distances (for specific topology use value 1: when this one is available; 0 not available). These characteristic indications can be output from another component: with graphic computing or with another NN (recurent). In this case object can be recognized no matter of their location, rotation or scale. It is very important to correct data like input. This case is used in this research.

Output of NN:

Topology and way of working of FFNN and SOM are not allowed same output. Here are given some output case, but everyone is applicable only for one of them:

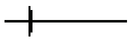
- Output Neurons number is the same as number of classes images: In this case only one output will be active (the output of the identified type) Its value will be around 1, all others should be 0. This is the most usual case in FFNN. Used in current research.
- One output neuron: In this case for every kind of image there is an interval. And depending of the interval could be decided if given image is from given class. Applicable to FFNN.
- Output neurons much more than classes images: In this case decision is taken depending on the distance to given neuron.

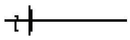
Topology of Hidden layers:

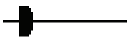
- FFNN: Usually used with one hidden layer, which make NN complicated and powerful. It’s possible to be realized without hidden layers. This is more simple and faster. To apply this case is necessary linear dividing between input and output (every output signal is linear depending on input signals). This is applicable for used images and because of this is used in current research.
- SOM: The Topology is without hidden layer.

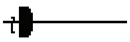
III. Results and Discussion

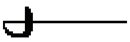
Used images:


foil with right (O1)


foil with pistol (O2)


epee with right (O3)


epee with pistol (O4)


sabre (O5)

Every epoch is from 500 images (input vector, output target vector).

The pictures which are used for this training are shown down. Characteristic indications are shown on Fig.3. The first and second are specific topology from the image (1 means that this topology is available; 0 mean not available). The third Characteristic indication is relationship between 2 distances.

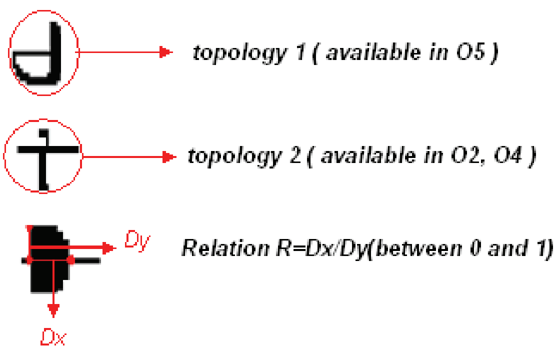


Fig. 3. Characteristic indications.

Results for FFNN:

Most researches are shown that neurons in hidden layer of FFNN should be as minimal as is possible for successful result. With a little number of neurons enough for successful training result will be much faster than with bigger number. In current research is possible without hidden layer, which reduce many calculation and remove all possibilities of local extremum. Of coarse it's not always possible to use this way, but it's possible to be checked at first without, then with hidden layers. A. Osareh [4] is found that 5 neurons are optimal for a given research tested between 1 and 30.

Following graphic on Fig. 4 is shown how the calculated outputs are moving depending from Epoch number. The output that should be active (Series1) maximum (Series2) and minimum (Series3) value of outputs that should be inactive (rest images).

After the first epoch values of output which should be active are around 0.2-0.25 the rest outputs are smaller. It is possible after 3 to 7 epochs the output which should be active to be smaller than one of the others. The reason is local extreme values. After 10 epochs the output which should be active are around 0.4-0.45the highest value. The rest are around 0.3.

Slowly output which should be active inclines to 1 all others to 0. After 50 epochs: values of outputs which should be active are around 0.9 almost 1. Rest are almost 0.

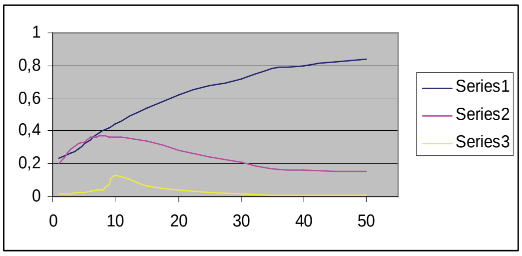


Fig. 4. Calculated output values of output that should be active (Series 1), minimum (Series 2) and maximum (Series 3) calculated outputs that should be inactive.

Results of SOM:

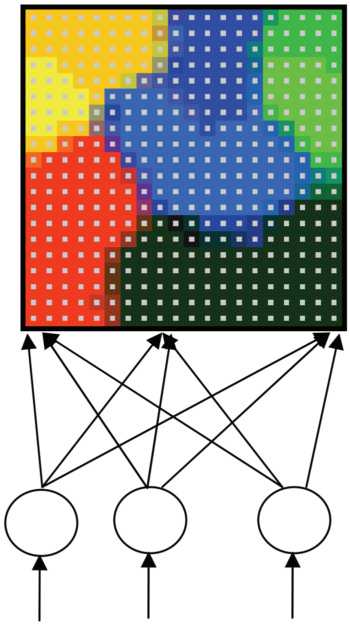


Fig.5 Scheme of using SOM and experimental result

Reyes-Aldasoro [7] uses SOM to decide if one image is from given type depending on the Euclidian distance between target and wining neuron. In given case there are 5 different types and it's not applicable. Because often the distance between winner and real target is bigger then distance between winner and fake target. This way of using will be applicable if we use five different SOMs (for all kind of weapon). This is out of the scope of this research.

After passing of one epoch all images from training set are grouped (clusterized) as can be seen in Fig 5. Let's accept that each of input parameters are passed like RGB and can be seen that all neurons are with different color. After each iteration boundaries between different groups become thinner. If iterations are not enough the images will not be clusterized correct. One possible way to use this approach for identification is to found the boundaries and the winner will be in same boundaries with the real target, but this require new application for founding boundaries. Another using is to see how images are divided in training set to find clusters: clusterization.

IV. CONCLUSION

In current research applications with 2 kind of NN are taken into account. They are use for non graphic computing of images. Used models of NN are FFNN with backpropagation and SOM. Results show that SOM is applicable for clusterization, FFNN for identification.

REFERENCES

[1] G. Gochev "Computer sight and neural networks", 1998

[2] http://www.ai-junkie.com/ann/som/som1.html

[3] B.WIDROW, FELLOW, IEEE, AND M. A. LEHR „30 Years of Adaptive Neural Networks: Perceptron, Madaline, and Backpropagation", PROCEEDINGS OF THE IEEE, vol. 78, no. 9, pp. 1415-1442, 1990

[4] A. Osareh , M. Mirmehdi , B. Thomas and R. Markhamb, "Automatic Recognition of Exudative Maculopathy using Fuzzy C-means Clustering and Neural Networks", Medical Image Understanding and Analysis, pp. 49-52, 2001

[5] M.N. Dailey, G.W. Cottrell "Organization of face and object recognition in modularneural network models", Neural Networks, vol. 12, no.7, pp. 1053–1074, 1999

[6] S. Lawrence, C. L. Giles, A. C. Tsoi, "What Size Neural Network Gives Optimal Generalization? Convergence Properties of Backpropagation", Technical Report UMIACS-TR-96-22 and CS-TR-3617, 1996

[7] C.C. Reyes-Aldasoro, A. L. Aldeco "Image segmentation and compression using neural networks", Advances Artificial Perception Robotics, 2000

[8] I. S. Bajwa, M. S. Naweed, M. N. Asif , S. I. Hyder "Feature Based Image Classification by using Principal Component Analysis", ICGST - Journal of Graphics Vision, and Image processing, vol. 09, no.2, pp 11-17, 2009