computing@computingonline.net
www.computingonline.net

ISSN 1727-6209
International Journal of Computing

AN IMPROVED ARCHITECTURE FOR COMPETITIVE AND COOPERATIVE NEURONS (CCNS) IN NEURAL NETWORKS

M. Kamrul Islam

School of Computing,
Queen's University,
Kingston, K7L 3N6, ON, Canada
islam@cs.queensu.ca

Abstract: *In neural networks, the associative memory is one in which applying some input pattern leads to the response of a corresponding stored pattern. During the learning phase the memory is fed with a number of input vectors and in the recall phase when some known input is presented to it, the network recalls and reproduces the output vector. Here, we improve and increase the storing ability of the memory model proposed in [1]. We show that there are certain instances where their algorithm can not produce the desired performance by retrieving exactly the correct vector. That is, in their algorithm, a number of output vectors can become activated from the stimulus of an input vector while the desired output is just a single vector. Our proposed solution overcomes this and uniquely determines the output vector as some input vector is applied. Thus we provide a more general scenario of this neural network memory model consisting of Competitive Cooperative Neurons (CCNs).*

Keywords: *Competitive cooperative neuron, associative memory, vector, frequency bands.*

1. INTRODUCTION

The ability to store and retrieve information is critical in any type of neural network. In neural network, the memory, particularly associative memory, can be defined as the one in which the input pattern or vector leads to the response of a corresponding stored pattern (output vector). That is, when an input vector is presented, the network recalls the corresponding output vector associated with the input vector. There are two types of associative memories: *autoassociative* and *heteroassociative* memory. In the case of autoassociative memory, both input and output vectors range over the same vector space. For example, a spelling corrector maps incorrectly spelled words (e.g. "matual") to correctly spelled words ("mutual"). Heteroassociation involves the mapping between input and output vectors over a different vector space. For example, given a name ("John") as input, the system will be able to recall its corresponding phone number ("657-9876") stored in memory.

In the context of neural network, an associative memory consists of neurons (known as conventional McCulloh-Pitts [2] neurons) that are capable of processing input vectors and recalling output vectors. These conventional model neurons use inputs from each source that are characterized by the amplitude of input signals. In this way each neuron can receive, process, and recall only one component of a memorized vector. Towards realizing the concept of associative memory, one of the commonly used techniques uses *correlation matrix memory* [3] which encodes all input and output vector pairs $\{y_k, x^T_k\}(k=1,2,3,...,n)$ into a correlation matrix, $M = \sum_{k=1}^{n} y_k x_k^T$. Later in the recall phase the matrix M is decoded to extract the output vector when the corresponding input vector is introduced to the network. The limitation of correlation matrix memory, in terms of memory capacity, is that it requires exactly *n* neurons to recall *n* components of a vector. In this paper we study the problem of increasing memory storage and recall capacity of a general associative memory and offer an idea that provides and ensures more storage and correct recall ability of the memory model (one layer Competitive Cooperative Neuron (CCN) network model) proposed in [1]. Our proposed method has an improved architecture of the CCN network where we need only N neurons (CCNs) to store and recall NR memories where R is the number of zones (defined later) of a CCN.

The organization of the paper is as follows. First in Section 2, we provide a general description of a CCN. In Section 3 we show how a network of such neurons can be formed and how they work by providing an example. Section 4 provides evidence to show the limitations found in the CCN network. Our result, that is, the improvement of the CCN network model is given in Section 5 which overcomes the shortcomings of the model [1]. Experimental results are described and shown in Section 6 followed by future research direction in Section 7. Finally, we conclude in Section 8.

2. DESCRIPION OF A CCN

Here we provide a concise description of the CCN which is the building block of a CCN neural network, the reader is referred to [1] for details. In order to increase the memory storage and the recall capacity of an associative memory compared to correlation matrix memory, the paper [1] introduces a novel type of model neuron called CCN as the building block of an associative memory. This model offers two new features: one is that the input signals are characterized by a two-dimensional parameter set representing the *amplitude* and the *frequency* of signals, whereas the conventional inputs have only component, the amplitude. The other feature of the CCN is that it consists of several distinct and autonomous receptor zones where each zone is able to receive a number of such input signals. Each zone is capable of selecting just one signal (called the *winning signal*) from the inputs signals it receives. A model of such a CCN is given in Figure 1.

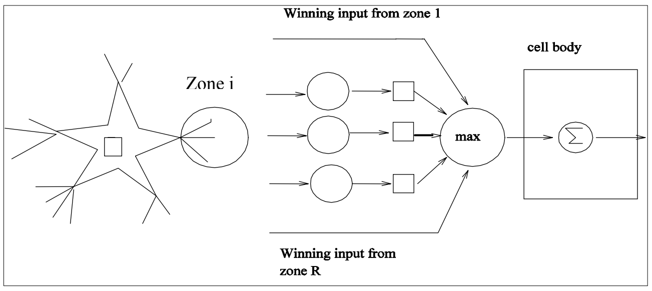


Fig. 1 – A CCN Model: the CCN on the left has five autonomous zones, each of which has a narrow bandwidth of frequencies that it can detect. Each zone receives m input signals. In each zone, only the input signals that have a frequency fi, r that falls within the zone's bandwidth participate in the competition and the winner is the signal with the highest effective amplitude. All the winning signals are propagated to the cell's body, where they cooperate and the cell is activated if the cumulative amplitude is greater than the cell's threshold.

The CCN consists of a number of zones R and each zone $r \in R$ collects input from many sources, $S(r) = \{S_1(r),S_2(r),S_3(r),...\}$. Each input signal $S_i(r) = (F_i(r),A_i(r))$ has two components which are normalized to one - the frequency $F_i(r) \in [0,1]$ which encodes the information [4] and the amplitude $A_i(r) \in [0,1]$-the strength of the signal. Each zone is sensitive to a small range of frequencies (also called *bands*) which means that any input signal whose frequency falls in the range can participate in the competition to be chosen by the zone.

The center of the band of input zone r of a CCN n at time t is denoted by B(n,r,t) and the tolerance level is T(n,r,t). The tolerance level is the dispersion from the center of the band which defines the range of frequency zone r can handle, that is, [B(n,r,t)-T(n,r,t),B(n,r,t)+T(n,r,t)]. The tolerance level and the center of band are not constant, they change over time. A zone can detect the input signals whose frequencies that are fall in the range of its frequencies and the amplitudes of these exceed a certain threshold value $\tau(n,r,t) \in [0,1]$. An input i in zone r wins if $A_i(r) >= \tau(n,r,t) > 0$ and $F_i(r) \in [B(n,r,t)-T(n,r,t),B(n,r,t)+T(n,r,t)]$. A zone is called active if it can select such an input signal. This way each zone propagates its winning signal to the cell body. Finally, the CCN fires if the combined amplitude of all the winning input signals from all the zones exceeds the threshold v(n,t) of the CCN body, that is, $\sum_{r=1}^{R} A_{i(w)}(r) \geq v(n,t)$ where $A_{i(w)}(r)$ is the amplitude of the winning signal of zone r. As the CCN is fired (activated) it sets the center of the frequency band of an active zone r to its corresponding winning signal, i.e., $B(n,r,t)=F_{i(w)}(r)$ where $F_{i(w)}(r)$ is the frequency of the winning signal of zone r. As the CCN fires, it generates the output vector whose components are the winning signals of all the active zones of the CCN. However, the output vector can be modified by using Hebbian learning [5] protocol. Initially, a CCN body threshold is set with the value v(n,t), which is greater than or equal to the sum of the zone thresholds, i.e., $v(n,t) \geq \sum_r \tau(n,r,t)$, so that in order for the CCN to fire, either all the zones must be active or some of them must receive a very strong signal to activate the CCN.

When fired, the CCN decreases tolerance levels of the corresponding active zones by some pre-specified value and the amplitude of the threshold of the CCN is also decreased to some level. Similarly if the CCN is not fired the corresponding tolerance levels of active zones are increased (anti-Hebbian learning). However, when the CCN fires we say the CCN specializes or learns. Thus by modifying the tolerance levels and the amplitude threshold of the CCN, the CCN is trained. In this way, the training phase stores an input vector whose components are the values of the frequencies of the winning signals which are set in the corresponding centers of the bands of the active zones. When it receives input in some but not all zones it uses that input to recall the previous inputs to the idle zones. For example [1], if a CCN has three zones and it fired when the input was the vector that represents the triple (Red, Sweet, Strawberry), then the next time it receives only "Red" and no input from the other zones, it will fire ("Red", "Sweet", "Strawberry") provided that the amplitude of the input signal ("Red") exceeds the cell's threshold.

2.1 CCN NETWORK MODEL

A simple one-layer feedforward network with three CCNs and three input sources is shown in Fig. 2. Each CCN has three receptor zones represented by the vertices of the triangles. The number of inputs to the different zones is not necessarily the same, but it can be made the same by adding zero-weight input signals.

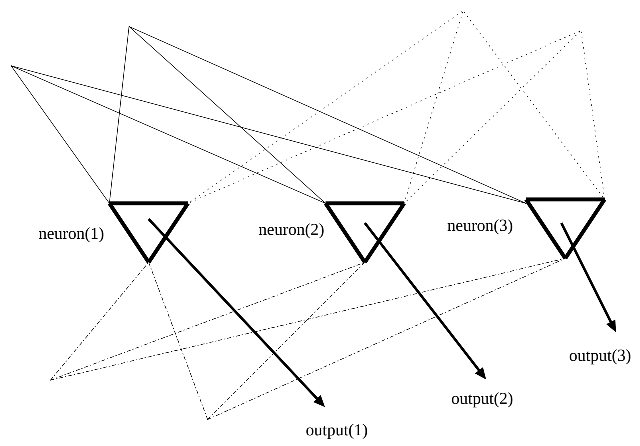


Fig. 2 – A simple one-layer feedforward network with three CCNs and three input sources is shown. Each CCN has three receptor zones represented by the vertices of the triangles.

3. HOW A CCN NETWORK WORKS

In order to understand the function of the network explained above, consider a one layer feed-forward network of CCN with N CCNs where each CCN $n \in N$ has 3 zones. Assume the centers of the frequency bands $B(n,r_1,t)$, $B(n,r_2,t)$, $B(n,r_3,t)$ of zones 1, 2 and 3 of CCN n are set to 0.150, 0.450, and 0.750 respectively. Let the tolerance level T for all zones be 0.050. The range of frequencies for each zone becomes [B-T, B+T], i.e., [0.100, 0.200], [0.400, 0.500] and [0.700, 0.800] and let the thresholds of zones 1, 2, and 3 be $\tau_1=0.20$, $\tau_2=0.15$, $\tau_3=0.15$ respectively and the CCN body threshold v=0.32

Assume that we want the network to store and recall vectors consisting of name, gender and id of students. Let the input vector be ("Sarah", "Female", "4781234") which is represented by frequencies (0.130, 0.416, 0.725). Let 0.20 be the amplitude of each of the components of the vector. As the input vector (assuming first component of the input vector to zone 1, second component to zone 2 and so on) is applied to the network, all the zones become active and the CCN starts firing since 0.20+0.20+0.20>0.32. If it does not fire then the tolerance level of inactive zones can be increased gradually to accommodate the frequency (anti Hebbian learning [5]). Now as the CCN fires the threshold to each zone is reduced to some minimum level (to some minimum value required to activate the corresponding zone) and the threshold to the cell body is also reduced to some minimum value. Let the cell body threshold be reduced to v=0.18. Now the center of the frequency band of each zone will be assigned the frequency of its winning signal, i.e., frequencies (0.130, 0.416, and 0.725) are assigned to zones 1, 2, and 3 respectively and the output vector becomes (0.130, 0.416, 0.725).

Now in the recall phase if we apply only input "Sarah" to zone 1 and no input from the other zones, the CCN will fire (because the threshold 0.20 of signal "Sarah" is greater than the CCN threshold 0.18) and produce the whole output vector (0.130, 0.416, 0.725) representing the vector ("Sarah", "Female", "4781234"). Therefore, if there are R (R-dimensional vector) zones in a CCN, then a single input signal to a zone will result in R recalled features (R-1, if we exclude the activating input) from the other zones, which is more efficient than recalling only one feature from every input compared with the correlation matrix memory. In general, if there are N CCNs each with R zones (i.e., a CCN can store and recall an R- dimensional vector) in a one-layer feed forward CCN network then it is able to recall total NR memories. This is because, after the training is complete a signal (as given in the previous example, only input component "Sarah") to a zone in a CCN will be strong enough to fire the CCN and recall all the other signals of other zones of that CCN. As a whole, only N input signals to N CCNs will suffice to recall NR memories. On the other hand, the correlation matrix memory needs NR CCNs to recall NR memories. Therefore the achievement of performance in terms of stored-features/number-of-CCNs ratio is higher in CCN network as compared to correlation matrix memory [3].

4. LIMITATIONS IN CCN NETWORKS

Here we consider a situation where the CCN model cannot achieve the performance as mentioned in the paper. First, we show that there are certain instances where the existing CCN model [1] fails to produce the expected output result. Then we offer an improvement to the architecture of the network model such that stipulated performance can be ensured. The associative network consisting of the proposed CCNs [1] functions well if only one input vector can be attracted to at most one CCN of the network during training. This is only possible when no two CCNs have all the centers of the frequency bands (B(n,r,t)) are equal. The network may suffer serious limitation in manipulating (storing and recalling) data when all the centers of the frequency bands of a CCN coincide with those of any other CCN. Mathematically this situation can be expressed as the following: if there are R zones of a CCN then there exist at least two CCNs n_i and n_j such that

$$B(n_i,r_1,t)=B(n_j,r_1,t),\\ B(n_i,r_2,t)=B(n_j,r_2,t),...,B(n_i,r_R,t)=B(n_j,r_R,t).$$

Under these circumstances, the network reaches a situation where the same input vector, M_j is stored in different CCNs. This is because the input, M_j stimulates and fires all those CCNs which have the same centers of frequency bands of their zones. Here we show how storage capacity decreases for the case stated above. As mentioned earlier, if the associative memory network has N CCNs and each CCN has R zones then we can store and recall N memory vectors (total NR memories) where each memory vector M_i consists of R-components. In this way, we can say this is equivalent to recall exactly NR memories in total. Let S be the number of CCNs that have the same centers of frequency bands of in their zones. This means that there is a memory vector M_s whose input can simultaneously fire S CCNs.

As M_s is stored in all the S CCNs, their centers of frequency bands will be assigned the corresponding frequencies of M_s(each component of M_s is represented by a frequency) and no other vector M_t, ($M_s \neq M_t$) can be stored in any of the S CCNs. So we have only N-S CCNs left to store N-1 vectors. If S>1, then we can not store all the remaining vectors (remaining N-1 vectors, since only one memory vector M_s is stored) to the memory. Therefore, this case does not allow us to store and recall N vectors. In the worst case, if all the N CCNs have the same centers of frequency bands then we can store only one vector in the whole network instead of N vectors. Thus the performance degrades down to 1/N percent which is quite worst for large values of N. In general, let S_1 be the number of CCNs having the same centers of frequency band $f^1_1,..., f^1_R$, S_2 be the number of CCNs with the same centers of frequency band $f^2_1,..., f^2_R$ and S_p be the number of CCNs with the same centers of frequency band $f^p_1,...,f^p_R$, then we can achieve p/N percent of vectors to be stored and recalled correctly where $S_1+S_2+...+S_p=N$. The following example demonstrates such a case.

Suppose the network is required to store input vectors {0,1,0,0}, {1,1,0,0}, {1,0,1,0} and recall when any of the vectors is presented to the network. Assume we have a network consisting of three CCNs each with four zones. Let the first, second, and third CCN's band centers are 0.1, 0.2, 0.1, 0.1; 0.1, 0.2, 0.1, 0.1 and 0.2, 0.2, 0.1, 0.1, respectively. Let 0 and 1 be encoded by the frequencies 0.1 and 0.2 respectively. Therefore, we obtain the equivalent representation of the four input vectors as {0.1, 0.2, 0.1, 0.1}, {0.2, 0.2, 0.1, 0.1} and {0.2, 0.1, 0.2, 0.1}, respectively. As we apply the input {0.1, 0.2, 0.1, 0.1} to activate some CCN, we find all the zones of CCNs 1 and 2 become active and they fire. Thus, the same input vector {0, 1, 0, 0} is stored in both CCNs. In this way, the two CCNs are stimulated and their centers of frequency bands are assigned the frequencies 0.1, 0.2, 0.1, and 0.1. When the second input pattern {0.2, 0.2, 0.1, 0.1} is presented it stimulates the third CCN and causes it to fire by storing the input frequencies to the corresponding centers of frequency bands. Now for the last input there is no CCN that can be activated since all the CCNs are already attracted to the two previous input vectors. Although we have three CCNs to store and recall three vectors according to the algorithm presented in the paper [1], we cannot store more than two input patterns in the associative memory for this particular example. Thus the performance of the proposed technique degrades in this case.

5. SOLUTION PROPOSED FOR CCN NETWORKS

In this section, we provide the improvement for the architecture of the CCN network to remedy the situation illustrated above. This is intended so that at most one CCN in the network can be stimulated (attracted) by a single input pattern. The modified network is shown in Fig. 3.

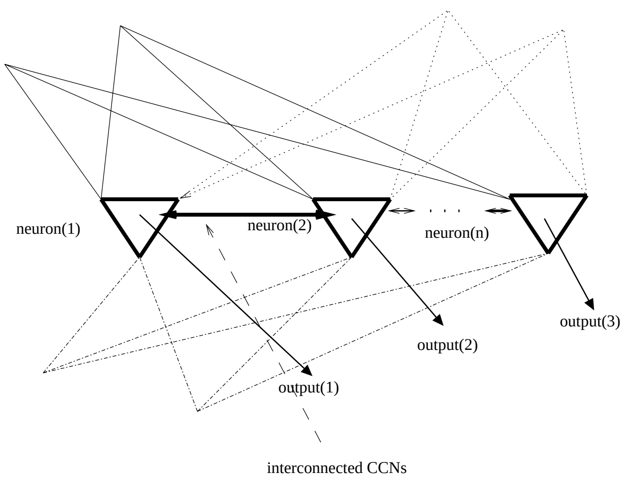


Fig. 3 – An improved simple one-layer feedforward network with three CCNs and three input sources is shown. CCNs are connected.

The main idea is to connect a CCN to its left and right neighbors and assign indices to them, except for the leftmost and rightmost CCNs which are only connected to their right and left CCNs, respectively. These indices, beginning from 1 to the number of CCNs, will be assigned arbitrarily among the CCNs. It is assumed that the CCN with the lowest index has the highest priority and priority will decrease with the increase of indices. After an input pattern is applied to the network, if a CCN gets stimulated (call it *active*) then it sends its index to all other CCNs. If it is not active then it refrains from sending its index. We ensure that the highest priority active CCN will be the one to be attracted to the input if there are more than one such active CCNs.

As each active CCN sends its index to all other, every active CCN compares the index it receives from other active CCNs and if any of the indices is smaller than its own index then it does not update its center of frequency band. This means that although it is a candidate for the input to store, it withdraws its candidacy and let other higher priority CCNs be attracted to the input. In this way, only the smallest indexed CCN wins and processes the input and changes its centers of frequency bands of its zones to the corresponding winning frequencies. Mathematically, this is a one-to-one function f: $\boldsymbol{S} \rightarrow$ N, where $\boldsymbol{S}$ denotes the set of CCNs in the network. Let $S \subseteq \boldsymbol{S}$ denote the set of CCNs simultaneously attracted to an input. It is obvious that there will be exactly one $C' \in S$ where $f(C') \neq f(C'')$ ($C'' \in S-\{C'\}$). By following the above procedure to propagate the indices among the CCNs, we obtain exactly one active C' which has the smallest index among the indices of the CCNs in S since f is one-to-one. For example, in a network of 11 CCNs, if CCNs with indices 2, 7, 11 become active for some input pattern, then the CCNs 7 and 11 will withdraw because they find the index 2 is smaller. As a result, CCN 2 will take over and become stimulated and attracted to the input. The introduction of priority ensures that at any time when an input pattern is presented in the network at most one CCN will be attracted to that input. Thus we eliminate the chance of firing more than one CCNs by a single input which overcomes the problem mentioned in earlier section.

6. EXPERIMENTAL RESULTS

According to the method presented in the previous section, we provide the numerical results and analyze the outcome by computing the percent of memory that can be stored and retrieved successfully. We compare the outcome of our results with the conventional network as proposed in [1]. There are two phases, namely, the training phase and the recall phase. In the training phase, we setup the network with a certain number of CCNs and the number of features or vector elements for each of the CCNs. After feeding the inputs in the network, we train the network to memorize the vectors in the CCNs. That is, the individual CCNs memorize the vectors by the technique mentioned above. The number of vectors to be stored equals the number of CCNs in the network and the number of elements in each vector is equal to the number of zones of each CCN. After we finish training the network, we perform the recall phase in which we provide only some element, m_i of a vector, $M=(m_1,m_2,...,m_R)$ to the network and the network produces the whole vector (i.e., all the elements of M).

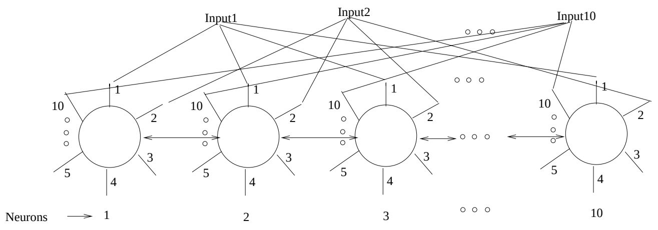


Fig. 4 – The CCN network model of our experiment is shown for 10 CCNs.

We perform our experiment with the network shown in Fig. 4 consisting of N=10 CCNs and each CCN has R=10 zones meaning each input vector $M=(m_1,m_2,...,m_{10})$ has 10 elements that can be accommodated by a single CCN. First, we remove the interconnections (denoted by the double arrow line segments between CCNs) among the CCNs so that the network becomes the one as described in [1] and then we perform the following experiments.

We have 10 sets of centers of frequency bands uniformly distributed over [0,1]. For example, the centers of frequency bands $B(n_i,r_1,t)$, $B(n_i,r_2,t)$,

$B(n_i,r_3,t),...,B(n_i,r_9,t)$, $B(n_i,r\_{10},t)$ for a CCN n_i can be 0.01,0.02,...,0.10 for zone 1,2,...,10, respectively. Thus, each set of frequency bands starts with 0.10*k+0.01 and ends with 0.10*k+0.10 where k=0,1,2,...,9. And the value of the tolerance level T for all zones is set to 0.005. In the experiment, randomly 10 sets of centers of frequency bands are assigned to 10 arbitrary CCNS.

Recall from the notation $S_1, S_2,...,S_p$ in Section 4 where S_i is the number of CCNs having the same centers of frequency band $f^i_1,..., f^i_{10}$, and $\sum_i S_i = 10$. We now give one sample of our experiment and then we provide the simulation results. In one random input for the network model, we find $S_1=2$, $S_2=1$, $S_3=1$, $S_4=3$, $S_5=1$, $S_6=1$, $S_7=1$ and $S_1+S_2+S_3+S_4+S_5+S_6+S_7=10$. This amounts to store and retrieve only seven (7) vectors $(S_1,S_2,..., S_7)$ of 10*7=70 vector elements in total by the network model of the experiment.

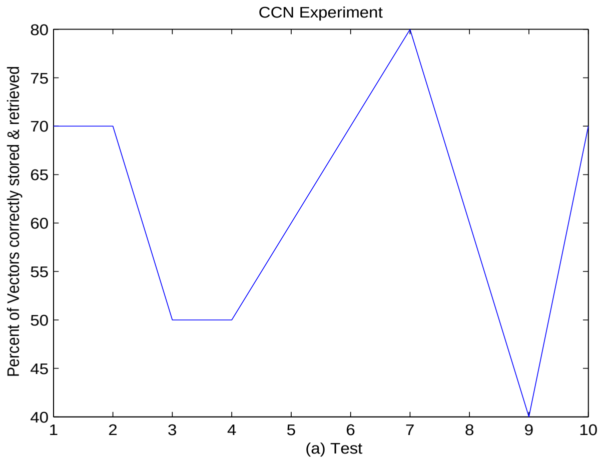


Fig. 5 – Showing result for a set of 10 runs where for each run the corresponding percentage of performance of correctly storing and retrieving vectors is shown along the y-axis.

This is because, for some S_i, $i \in \{1,2,...,7\}$ we can store only one vector (consisting of 10 elements) instead of all 10 vectors of S_i*R elements whereas our proposed method can store and retrieve all N=10 vectors of N*R=10*10=100 vector elements in total. Thus in this particular instance the performance of conventional method in terms of storing and retrieving vectors of around 70 percent of the total vectors.

Fig. 5 shows the graph for a set of 10 test runs where the test numbers are put along the x-axis and the corresponding percentage of vectors that can be stored and retrieved uniquely are shown in the y-axis. This random set of 10 test runs gives us an overview of how the CCN network [1] performs. Of the 10 runs we can achieve 80 percent performance once, i.e., eight vectors out of 10 are stored and retrieved correctly (on the 9th test run). And in general, the performance lies around 70 percent for a number of times and shows poor percentage (40 percent) once.

However, these random runs do not tell us about the true characteristics of the conventional CCN network. Thus we perform 100 runs and then average the results of the 100 runs which is depicted in Figure 6. Here we put the value of percentage of correctly storing and retrieving vectors in the x-axis and the number of times (out of 100 runs) the corresponding percentage is obtained is shown in the y-axis. This gives us a clear view of the characteristics of the CCN network [1].

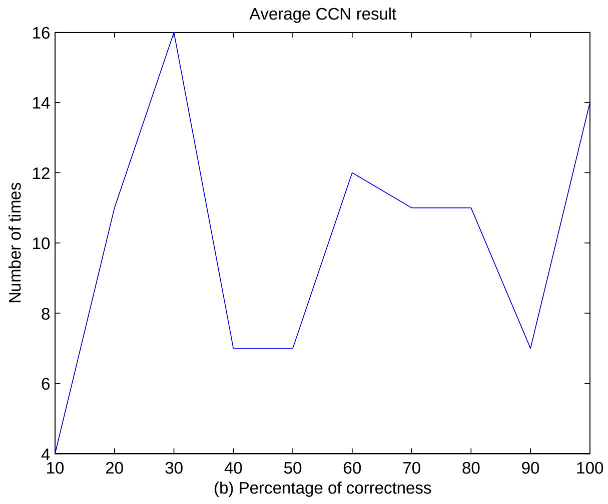


Fig. 6 – The average result of 100 runs where the percentage of performance is shown in the x-axis and the number of times the corresponding percentage achieved is shown in the y-axis.

We observe that full percentage (100%) of performance is achieved only 14 times (out of 100 times) and the low (10%) of performance is achieved only 4 times (out of 100 times). That means, we could excite or fire all (10) the CCNs in our experiment 14 times and only one CCN in 4 times for storage and retrieval of vectors. But these irregularities are alleviated by our proposed method when we perform experiments with the interconnections among the CCNs.

In order to get a broader view of how the system works for large networks, we changed the number of CCNs to N = 1000. We also have different values for the number of zones, namely, R has values in {10, 12, 14, 16, 18, 20, 22}. As can be understood by the intuition that the higher the number of zones, the fewer the error rate of correctly retrieving stored vectors. This is because, if the number of zones is higher then it will be less likely that two or more zones will have the same set of winning signals. The algorithm in [1] does better when the number of zones in each CCN is higher. Figure 7 shows the graph for a network of 1000 CCNs where in the x-axis we have the values of different R and the y-axis shows the percentage of vectors correctly retrieved and stored by their algorithm [1]. The graph shows that as the number of zones increases the percentage of vectors manipulated (i.e., stored and retrieved) increases and vice versa. It can be observed from the graph that in almost all cases their algorithm achieves more than (96%) success. However, applying our algorithm results in correctly manipulating all the vectors. For each value of R, we conducted 100 runs to figure out the overall of the performance. Figure 8 shows the average results for different values of R. The performance of the average result is quite stable.

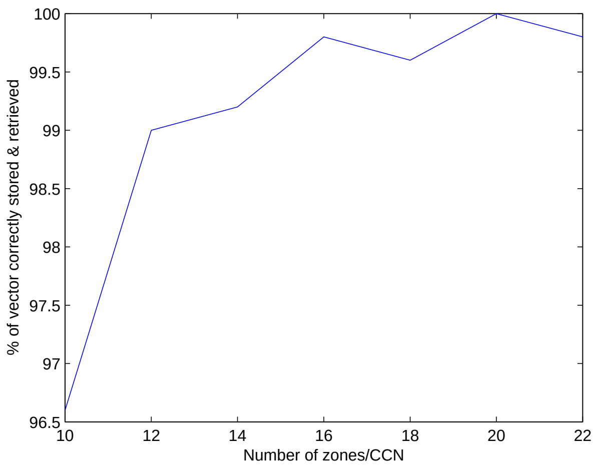


Fig. 7 – The graph shows the relationship between the number of zones and the percentage of correctly storing and retrieving the number of vectors.

7. FUTURE RESULTS

An improved and more general one-layer feedforward CCN network (more precisely associative memory network) depending upon the work of [1] has been introduced in the paper which can store R-dimensional input patterns in each of its neuron (each neuron has R-zones) and retrieve the whole vector with the presence of only a component of the input. We can further investigate the possibility of using recurrent network consisting of CCNs as recurrent network is well known and commonly used in the realm of neural network. In recurrent networks, the vector output of each of the neurons is fed back to that neuron's input lines, where each element of the output vector is fed back only to its corresponding zone. This ability may help us store and recall higher-order memories. Higher-order memory is one in which certain component of a vector can associate components or features with other vector. For example, consider the previous input pattern ("Sarah", "Female", "4781234"). After the network is trained, if we apply only the input "Sarah" we retrieve the whole vector ("Sarah", "Female", "4781234") as output. In the case of recurrent network we may feed the component "4781234" back to the appropriate zone which might result in recalling another vector with elements *School of computing*, *Queen's University*, *Canada* representing her department name, university name and the country where she is from.

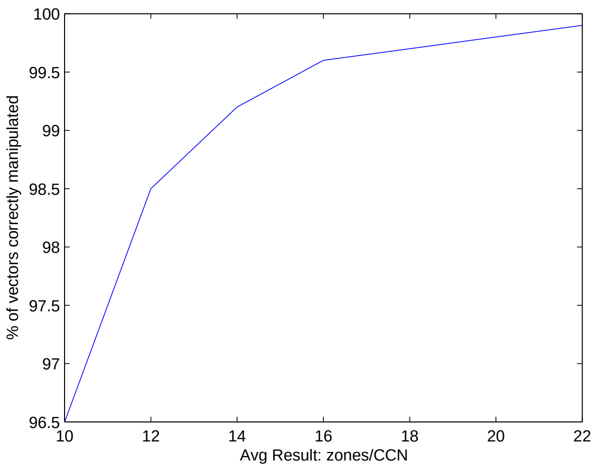


Fig. 8 – For each value of R, 100 runs were performed. y-axis shows the average of the percentage of manipulating vectors for each value of R.

Another important direction of this research could be identifying the data types of the elements of vectors and manipulating the data accordingly. Currently, we do not have the mechanism to separate different data types, namely identifying between string and integer types. It would be interesting if we could incorporate the data type separating mechanism in the network.

8. CONCLUSION

Motivated by the resemblance of a pyramidal cell [6] found in brain, the authors of [1] proposed a new type of model neuron (called CCN) to imitate the behavior of a pyramidal cell. The pyramidal cell is believed [7] to process both the frequency and the amplitude of the input signals and there is some sort of competition among inputs. Attempts are made to follow the physical structure and functional behavior of the pyramidal cell to some extent in the CCN, such as competition among the inputs and finally select the winner. In this paper, we provide an improvement to the CCN model [1] which is more generalized and can handle situation where there is a possibility of getting activated more than one CCN. Furthermore, we can also increase the memory with our proposed modification to the architecture of the CCN. Thus the modified neuron model can increase the memory capacity substantially as demonstrated in this paper.

9. REFERENCES

[1] H. Bar. W. Miranker. A. Ambash. Competition and Cooperation in neural processing. *IEEE Transactions on Neural Networks* 53 (3) (2004).

[2] S. Haykin. *Neural Networks, a Comprehensive Foundation*. Upper Saddle River, NJ. Prentice-Hall, 1999.

[3] http://www.wikipedia.org/.

[4] J. Singh. *Great Ideas in Information Theory, Language and Cybernetics*. New York, Dover, 1966.

[5] D. Hebb. *The Organization of Behavior: A Neuropsychological Theory*. New York, Wiley, 1949.

[6] M. Arbib. *The Metaphoricalbrain*. New York: Wiley-Interscience, 1972.

[7] L. Rutherford. S. Nelson. G. Turrigiano. BDNF has opposite effects on the quantal amplitude of pyramidal neuron and interneuron exciatory synapses. *Neuron*, 21, (1998), p. 521-530

***Kamrul Islam** is currently pursuing Ph.D in the School of Computing in Queen's University, Kingston, Ontario, Canada where he obtained his Master's degree in 2005. His research interests include algorithm designs, complexity analysis in the field of Computational Geometry, Sensor Networks and Neural Networks.*

computing@computingonline.net
www.computingonline.net

ISSN 1727-6209
International Scientific
Journal of Computing

FORMING EVOLUTIONARY DESIGN OF NEURAL NETWORKS WITH DIFFERENT NODES

Eva Volna

University of Ostrava,
30th Dubna st. 22,
701 03 Ostrava, Czech Republic
e-mail: eva.volna@osu.cz, http://www.osu.cz

Abstract: *Evolution in artificial neural networks (e.g. neuroevolution) searches through the space of behaviours for a network that performs well at a given task. Here is presented a neuroevolution system evolving populations of neurons that are combined to form the fully connected multilayer feedforward neural network with fixed architecture. In this article, the transfer function has been shown to be an important part of architecture of the artificial neural network and have significant impact on an artificial neural network's performance. In order to test the efficiency of described method, we applied it to the pattern recognition problem and to the alphabet coding problem.*

Keywords: *Neuroevolution, multilayer feedforward network, pattern recognition problem, alphabet coding problem.*

1. NEUROEVOLUTION

Neuroevolution represents a combination of neural networks and evolutionary algorithms (e.g. the genetic algorithm) where neural networks are the phenotype being evaluated. The genotype is a compact representation that can be translated into an artificial neural network. Evolution has been introduced into artificial neural networks at roughly three different levels: connection weights, architectures, and learning rules. The evolution of connection weights provides a global approach to connection weights training, especially when gradient information of the error function is difficult or costly to obtain. Due to the simplicity and generality of the evolution and the fact that gradient-based training algorithms often have to be run multiple times in order to avoid being trapped in a poor local optimum, the evolutionary approach is quite competitive. The evolution of architectures enables artificial neural networks to adapt their topologies to different tasks without human intervention and thus provides an approach to automatic artificial neural network design. Simultaneous evolution of artificial neural network architectures and connection weights generally produces better results. The evolution of learning rules in artificial neural networks can be used to allow an artificial neural network to adapt its learning rule to its environment. In a sense, the evolution provides artificial neural network with the ability of learning to learn.

Evolutionary algorithms are the term for different approaches as of using the models of evolutionary processes, which have nothing common with biology. They try to use the conception of driving forces of organism's evolution for optimization purposes. Evolutionary algorithms refer to a class of population-based stochastic search algorithms that are developed from ideas and principles of natural evolution. Fogel [1] gives a good introduction to various evolutionary algorithms for optimization. One important feature of all these algorithms is their population-based search strategy. Individuals in a population compete and exchange information with each other in order to perform certain tasks.

The choice of the right representation of individuals and their fitness create the essence of the advantageousness of the evolutionary algorithm, which depends on the selection of suitable choice of evolutionary algorithm and its appropriate operators. Individual within the evolutionary algorithm are then the problem solution. If a new solution is better, it substitutes the previous one. Optimization will be considered here as a synonym for minimization [2]. This is not a problem because of going in search the function maximum is equivalent to going in search of function minimum multiplied by -1.

Global search procedures such as evolutionary algorithms are usually computationally expensive. It would be better not to employ evolutionary algorithms at all three levels of evolution. It is, however, beneficial to introduce global search at some levels of evolution, especially when there is little prior knowledge available at that level and the performance of the artificial neural network is required to be high, because the trial-and-error or heuristic methods are very ineffective in such circumstances. With the increasing power of parallel computers, the evolution of large artificial neural networks becomes feasible. Not only can such evolution discover possible new artificial neural network architectures and learning rules, but it also offers a way to model the creative process as a result of artificial neural network's adaptation to a dynamic environment.

2. EVOLUTION OF NODE TRANSFER FUNCTIONS

The architecture of artificial neural network includes its topological structure, i.e., connectivity, and the transfer function of each neuron in the artificial neural network. Architecture design is crucial in the successful application of artificial neural networks because the architecture has significant impact on a network's information processing capabilities. Up to now, architecture design is still very much a human expert's job. It depends heavily on the expert experience and a tedious trial-and-error process. There is no systematic way to design a near-optimal architecture for a given task automatically. Design of the optimal artificial neural network architecture can be formulated as a search problem in the architecture space where each point represents some architecture. Given some performance (optimality) criteria about architectures (e.g., lowest training error, lowest network complexity), the performance level of all architectures forms a discrete surface in the space. The optimal architecture design is equivalent to finding the highest point on this surface.

The discussion on the evolution of architectures so far only deals with the topological structure of architecture. The transfer function of each node in the architecture has been usually assumed that is fixed and predefined by human experts, at least for all the nodes in the same layer.

In principle, transfer functions of different neurons in artificial neural networks can be different (e.g. hard-limiting threshold function, a Gaussian function, sigmoid functions etc.) and decided automatically by an evolutionary process, instead of assigned by human experts. The decision on how to encode transfer functions in chromosome depends on how much prior knowledge and computation time is available. In general, neurons within a group, like a layer, in an artificial neural network tend to have the same type of transfer function with possible difference in some parameters, while different groups of neurons might have different types of transfer function. This suggests some kind of indirect encoding method, which lets developmental rules to specify function parameters if the function type can be obtained through evolution, so that more compact chromosomal encoding and faster evolution can be achieved. Little work has been only done on the evolution of node transfer function up to now. Mani proposed a modified backpropagation, which performs gradient descent search in the weight space as well as the transfer function space [3], but connectivity of artificial neural networks was fixed. Lovel and Tsoi investigated the performance of neocognitrons with various S-cell and C-cell transfer functions, but did not adopt any adaptive procedure to search for an optimal transfer function automatically [4]. Stork et al. [5] were the first to apply evolutionary algorithms to the evolution of both topological structures and node transfer functions even though only simple artificial neural networks with seven nodes were considered. The transfer function was specified in the structural genes in their genotypic representation. It was much more complex than the usual sigmoid function because authors in [5] tried to model biological neurons. White and Ligomenides [6] adopted a simpler approach to the evolution of both topological structures and node transfer functions. For each individual (i.e. the artificial neural network) in the initial population, 80% nodes in the artificial neural network used the sigmoid transfer function and 20% nodes used the Gaussian transfer function. The evolution was used to decide the optimal mixture between these two transfer functions automatically. The sigmoid and Gaussian transfer function themselves were not evolvable. No parameters of the two functions were evolved. Liu and Yao [7] used evolutionary programming to evolve artificial neural networks with both sigmoidal and Gaussian nodes. Rather than fixing the total number of nodes and evolve mixture of different nodes, their algorithm allowed growth and shrinking of the whole artificial neural network by adding or deleting a node (either sigmoidal or Gaussian). The type of node added or deleted was determined at random. Authors in [8, 9, 10] went one step further. They evolved topology of artificial neural network, node transfer function, as well as connection weights for projection neural networks. Sebald and Chellapilla [11] used the evolution of node transfer function as an example to show the importance of evolving representations. Representation and search are the two key issues in problem solving. Co-evolving solutions and their representations may be an effective way to tackle some difficult problems where little human expertise is available. In principle, the difference in transfer functions could be as large as that in the function type, e.g. that between a hard limiting threshold function and Gaussian function, or as small as that in one of parameters of the same type of function, e.g. the slope parameter of the sigmoid function. One point worth mentioning here is the evolution of both connectivity and transfer functions at the same time [5] since they constitute a complete architecture. Encoding connectivity and transfer functions into the same chromosome makes it easier to explore nonlinear relations between them. Many techniques used in encoding and evolving connectivity could equally be used here.

The evolutionary approaches discussed so far in designing artificial neural network architecture evolve architectures only, without any connection weights. Connection weights have to be learned after a near-optimal architecture is found. This is especially true if one uses the indirect encoding scheme of network architecture. One major problem with the evolution of architectures without connection weights is *noisy fitness evaluation* [18]. In other words, fitness evaluation is very inaccurate and noisy because a phenotype's (i.e., an artificial neural network with a full set of weights) fitness was used to approximate its genotype's (i.e., an artificial neural network without any weight information) fitness. We want to optimize the genotype so that it can perform well regardless of initial connection weights, but we can only approximate such optimization by examining phenotypes with limited sets of initial connection weights of a virtually indefinite number of sets. There are two major sources of noise [7]:

- The *first* source is the random initialization of the weights. Different random initial weights may produce different training results. Hence, the same genotype may have quite different fitness due to different random initial weights used in training.
- The *second* source is the training algorithm. Different training algorithms may produce different training results even from the same set of initial weights. This is especially true for multimodal error functions. For example, backpropagation may reduce an artificial neural network's error to 0.05 through training, but an evolutionary algorithm could reduce the error to 0.001 due to its global search capability.

In order to reduce such noise, an architecture usually has to be trained many times from different random initial weights. The average result is then used to estimate the genotype's mean fitness. This method increases the computation time for fitness evaluation dramatically. It is one of the major reasons why only small artificial neural network were evolved in this way. In essence, the noise is caused by the one-to-many mapping from genotypes to phenotypes. It is clear that the evolution of architectures without any weight information has difficulties in evaluating fitness accurately. One way to alleviate this problem is to evolve artificial neural network architectures and connection weights simultaneously [16, 19]. In this case, each individual in a population is a fully specified artificial neural network with complete weight information. Since there is a one-to-one mapping between a genotype and its phenotype, fitness evaluation is accurate.

3. FIXED-TOPOLOGY NEUROEVOLUTION

Fixed-topology methods require a human to decide the right topology for a problem and most of them optimize connection weights only. Some highest performing neuroevolution systems (e.g. a system that performs better than any other systems on benchmark tasks) realize a more parametric evolution. The Symbiotic Adaptive Neuro-Evolution algorithm (SANE) [12] deals with a population of individual neurons, each of which is represented by a numerical vector defining the weight of its connections to each of the input and output neurons. In addition, there is a population of network blueprints consisting of pointers to neurons in the population. Individual networks are built out of neurons' subsets that are specified by one of blueprints, and the performance of the network as a whole is assigned to both the blueprint and to each individual neuron contributing to the network. The network blueprints are also evolved, with crossover recombination between network representations, mutations where half of the pointers are changed to offspring of the neurons to which they previously pointed, and mutations where a small fraction of the pointers are set to completely random neurons. Similarly, the top performing neurons themselves are also recombined and mutated to produce new neurons. The Enforced Sub-Population (ESP) [13], variant of SANE, is based on the special sort of separation. Rather than having one large pool of neurons with network blueprints, ESP maintains a separate population of neurons for each position in the network. Building a network then consists of selecting exactly one neuron from each of these subpopulations. Since the populations are kept separate during the creation of the next generation, each population is able to focus on a particular function more quickly, and networks are less likely to have redundant neurons. Recently, in [14] is successfully applied a special evolutionary strategy called Evolution Strategy with Covariance Matrix Adaptation (CMA ES) to the evolution of fixed-topology neural networks. This method keeps track of correlations between changes of different weights in the network and fitness. Based on this information, the CMA-ES changes the covariance matrix of the weight mutation distribution so that it becomes more biased towards what were so far the most promising directions of search.

In the article, the transfer function has been shown to be an important part of architecture of the artificial neural network, one has significant impact on artificial neural network's performance. Here is presented a neuroevolution system evolving populations of neurons that are combined to form the fully connected multilayer feedforward neural networks with fixed architecture. Neuroevolution evolves transfer functions of each neuron in hidden and output layers of the network. The system maintains diversity in the population, because a dominant neural phenotype is likely to end up in the same network more than once. As several different types of neurons are usually necessary to solve a problem, networks with too many copies of the same neuron are likely to fail. The dominant phenotype then loses fitness and becomes less dominant. The system works well because it makes sure neurons get the credit they deserve, unlike some other neuroevolution techniques, where bad neurons can share in a good network or good neurons can be brought down by their network. It also works by decomposing the task, breaking the search into smaller, more manageable parts.

In the following is described a method of automatic search the node transfer function architecture in multilayer feedforward neural network: First, we must propose neural network architecture before the main calculation. We get the number of input (*m*) and output (*o*) neurons from the training set. Next, we have to define the number of hidden neurons (*h*) that is very confounding issue, because it is generally more difficult to optimize large networks than small ones. Thereafter the process of evolutionary algorithms is applied. Chromosomes are generated for every individual from the initial population as follows, see Fig. 1:

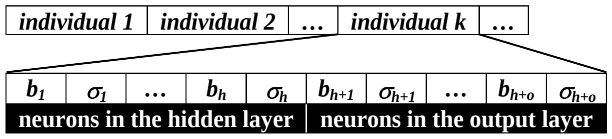


Fig. 1 – Population of individuals and their chromosomes.

Symbols b_i ($i = 1, \ldots, h+o$) refers to varies types of activation functions [15]:

- $b_i = 1$, if the activation function is a *binary sigmoid function*:

$$f(x) = \frac{1}{1+\exp(-\sigma\, x)} \tag{1}$$

where σ is the steepness parameter, which value is set in the initial population randomly, e.g. $\sigma_i \in \{1,2,3,4,5,6,7\}$.

- $b_i = 2$, if the activation function is a binary *step function with threshold* θ:

$$f(x) = \begin{cases} 1 & if\ x \geq \theta \\ 0 & if\ x < \theta \end{cases} \tag{2}$$

the steepness parameter σ_i is not define here thus we assigned value 0 to it.

- $b_i = 3$, if the activation function is a *Gaussian function*:

$$f(x) = \exp(-x^2) \tag{3}$$

the steepness parameter σ_i is not define here thus we assigned value 0 to it.

- $b_i = 4$, if the activation function is a *saturated linear function*:

$$f(x) = \begin{cases} 0 & if\ x < 0 \\ x & if\ 0 \leq x \leq 1 \\ 1 & if\ x > 1 \end{cases} \tag{4}$$

the steepness parameter σ_i is not define here thus we assigned value 0 to it.

Next, we calculate an error value (*E*) between the desired and the real output after defined partial training with genetic algorithms. Adaptation of each individual starts with randomly generated weight values that are the same for each neural network in the given population. On the basis of it is calculated a fitness function for every individual as follows:

$$Fitness_i = E_{max} - E_i. \tag{5}$$

for $i = 1, \ldots, N$; where E_i is error for the *i-th* network after a partial adaptation, E_{max} is a maximal error for the given task, $E_{max} = o \times pattern$ (*o* is number of output neurons and *pattern* is number of patterns), *N* is the number of individuals in the population.

All of the calculated fitness function values of the two consecutive generations are sorted descending and the neural network representation attached to the first half creates the new generation. For each fitness function is calculated the probability of reproduction its existing individual by standard method. One-point *crossover* was used to generate two offsprings. If the input condition of *mutation* is fulfilled (e.g. if a random number is generated, that is equal to the defined constant), one of the individuals is randomly chosen. There is randomly replaced one place in its genetic representation by a random value from the set of permitted values. The process of the evolutionary algorithm is ended, if the saturation parameter τ[1] is greater then define value, i.e. the population is composed only from similar types of individuals.

4. EXPERIMENTAL TASKS

In order to test the efficiency of described method, we applied it to the pattern recognition problem and to the alphabet coding problem that exists in cryptography.

Pattern recognition problem: For coding input examples the scheme from Fig. 2 is needed. If in an appropriate place the connection exists (e.g. on side with number 1, ..., 7), than the input chain representation is 1. If a connection does not exist, we have 0 in the appropriate place. Input vector contains thus seven bits. Output vector has got four bits and codes binary values of the input chain number representation.

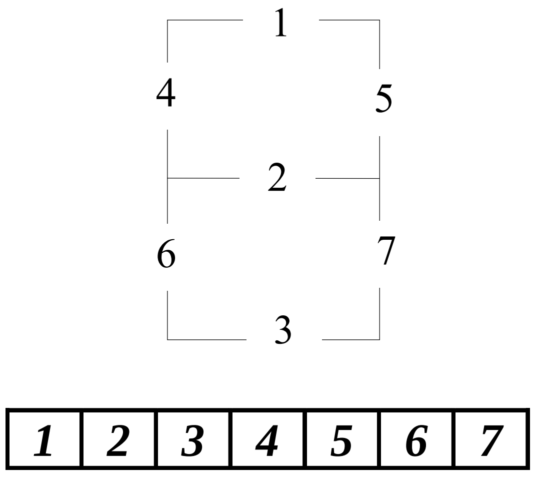


Fig. 2 – Scheme for coded examples to input chain of bits.

Neural network was trained to the following examples (see Fig. 3):

1011111→0000
0000101→0001
1110110→0010
1110101→0011
0101101→0100
1111001→0101
1111011→0110
1000101→0111
1111111→1000
1111101→1001

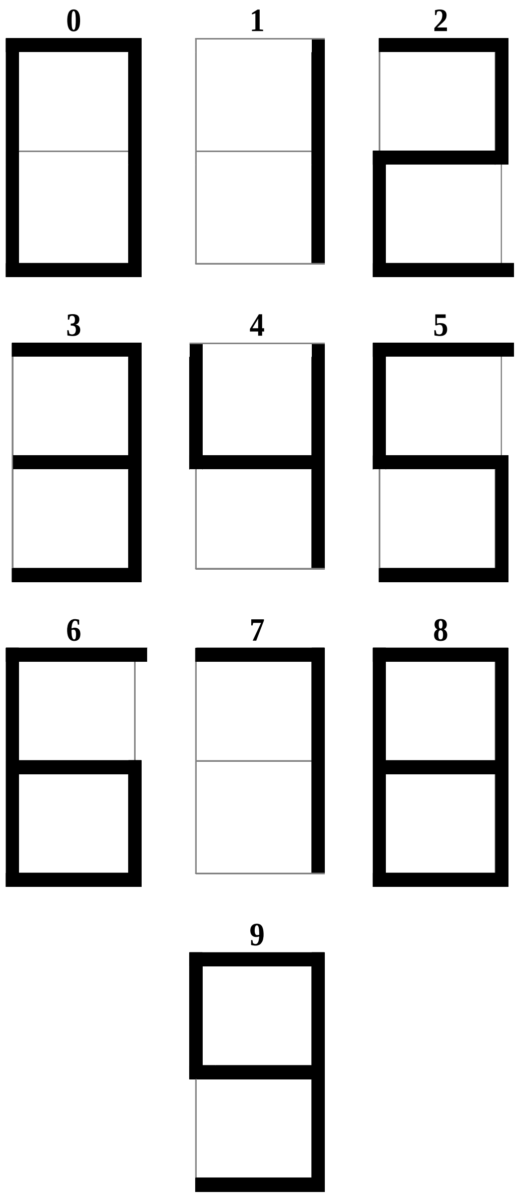


Fig 3 – The set of patterns (the training set).

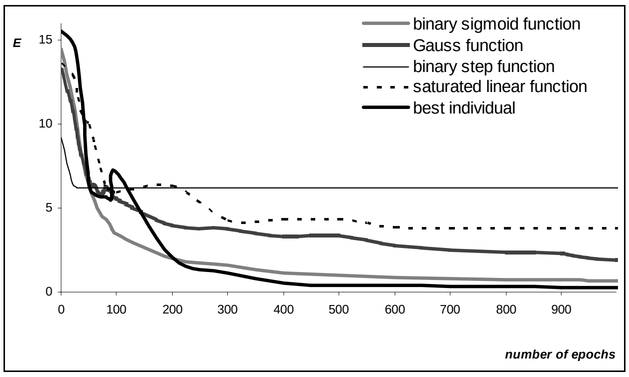
[1] τ is equal to a number of the same values at the same positions in chromosomes.

Table 1. The set of patterns (the training set).

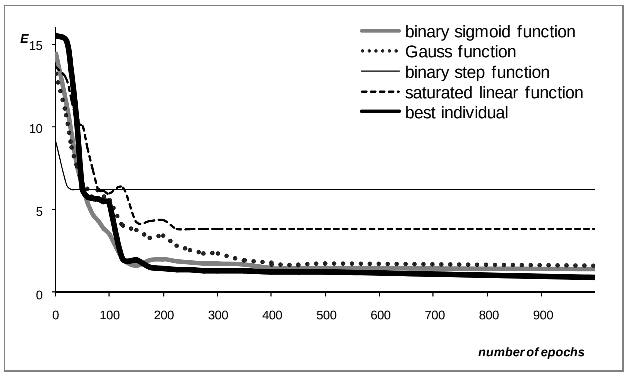
THE PLAIN TEXT			THE CIPHER TEXT
Char	**ASCII code (DEC)**	**The chain of bits**	**The chain of bits**
a	**97**	**00001**	**000010**
b	**982**	**00010**	**100110**
c	**99**	**00011**	**001011**
d	**100**	**00100**	**011010**
e	**101**	**00101**	**100000**
f	**102**	**00110**	**001110**
g	**103**	**00111**	**100101**
h	**104**	**01000**	**010010**
i	**105**	**01001**	**001000**
j	**106**	**01010**	**011110**
k	**107**	**01011**	**001001**
l	**108**	**01100**	**010110**
m	**109**	**01101**	**011000**
n	**110**	**01110**	**011100**
o	**111**	**01111**	**101000**
p	**112**	**10000**	**001010**
q	**113**	**10001**	**010011**
r	**114**	**10010**	**010111**
s	**115**	**10011**	**100111**
t	**116**	**10100**	**001111**
u	**117**	**10101**	**010100**
v	**118**	**10110**	**001100**
w	**119**	**10111**	**100100**
x	**120**	**11000**	**011011**
y	**121**	**11001**	**010001**
z	**122**	**11010**	**001101**

Alphabet coding problem: Neural networks can be also used in encryption or decryption algorithms, where parameters of adapted neural networks are included to cipher keys. Cipher keys must have several heavy attributes. The best one is the singularity of encryption and cryptanalysis [17]. Encryption is a process in which we transform the open text (e.g. news) to cipher text according to rules. Cryptanalysis of the news is the inverse process, in which the receiver of the cipher transforms it to the original text. The open text is composed from alphabet characters, digits and punctuation marks. The cipher text has usually the same composition as the open text. We worked [15] with multilayer neural networks, which topologies were based on the training set (see Table 1). The chain of chars of the plain text in a training set is equivalent to a binary value that is 96 less than its ASCII code. The cipher text is then a random chain of bits.

The initial population in both experiments contains 30 three-layer feedforward neural networks. Each network architecture is 7 - 7 - 4 for *pattern recognition problem*, and 5 - 5 - 6 for *alphabet coding problem* [15], because both problems are not linearly separable and therefore we cannot use neural network without hidden layer of neurons. All nets are fully connected. We use the genetic algorithm with the following parameters: probability of mutation is 0.01 and probability of crossover is 0.5. The saturation parameter τ is 95%. Adaptation of each neural network in given population starts with randomly generated weight values that are the same for each neural network in the population. We also used genetic algorithms with the same parameters for the partial neural network adaptation, where number of generations for a partial adaptation was 500. Their chromosome representation is described in [16].



(a)



(b)

Fig. 4 – The Error function history: (a) ***pattern recognition problem,*** **(b)** ***alphabet coding problem.***

History of the error functions is shown in the Fig. 4. There are shown average values of error functions in the given population. Other numerical simulations gave very similar results. The "*binary* *sigmoid function*" represents an average value after adaptation with the binary sigmoid activation function consecutively with all steepness parameters $\sigma = \{1,2,3,4,5,6,7\}$. The "*binary step function*" represents an adaptation with the binary step activation function (with the threshold θ), the "*saturated linear function*" represents an adaptation with the saturated linear activation function, and "*Gaussian activation*" represents an adaptation with the Gaussian activation function. Each of these mentioned representations is associated with all neurons in given neural network architecture. Opposite of this, the "*best individual*" represents an adaptation of the best individual in population, which chromosomes are the following, see Fig. 5:

b_1	σ_1	b_2	σ_2	b_3	σ_3	b_4	σ_4	b_5	σ_5	b_6	σ_6	b_7	σ_7
1	7	1	5	1	1	1	5	3	0	3	0	1	3

neurons in the hidden layer

b_8	σ_8	b_9	σ_9	b_{10}	σ_{10}	b_{11}	σ_{11}
3	0	1	7	1	2	1	6

neurons in the output layer

(a)

b_1	σ_1	b_2	σ_2	b_3	σ_3	b_4	σ_4	b_5	σ_5
1	5	1	7	1	1	3	0	1	5

neurons in the hidden layer

b_6	σ_6	b_7	σ_7	b_8	σ_8	b_9	σ_9	b_{10}	σ_{10}	b_{11}	σ_{11}
3	0	1	7	3	0	1	5	1	6	1	2

neurons in the output layer

(b)

Fig. 5 – The "best individual" chromosome in the last population: a) *pattern recognition problem,* (b) *alphabet coding problem.*

5. CONCLUSIONS

All networks solve the pattern recognition task resp. alphabet coding problem in our experiments, but artificial neural network with evolving transfer functions of each neuron works well, because several different types of neurons are usually necessary to solve a problem. We can see that the proposed technique is really efficient for the presented purpose, see the Fig. 3 or Table 1. Networks with too many copies of the same neuron work usually worse, see the Fig. 4.

Here, the transfer function is shown to be an important part of architecture of the artificial neural network and have significant impact on artificial neural network's performance. Transfer functions of different neurons can be different and decided automatically by an evolutionary process, instead of assigned by human experts.

In general, nodes within a group, like layer, in an artificial neural network tend to have the same type of transfer function with possible difference in some parameters, while different groups of nodes might have different types of transfer function.

6. REFERENCES

[1] D. B. Fogel. *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence.* IEEE Press. New York, 1995.

[2] P. Hammerstein, E. H. Hagen, A.V. Herz, M.H. Herzel. Robustness: A key to evolutionary design. *Biol. Theory* 1(1) 90–93, 2006.

[3] G. Mani. Learning by gradient descent in function space. *Proc. of the IEEE Int. Conf. "System, Man, and Cybernetics"*, Los Angeles, CA, 1990, pp. 242–247.

[4] D. R. Lovell. A. C. Tsoi. *The Performance of the Neocognitron with Various S-Cell and C-Cell Transfer Functions*, Intell. Machines Lab., Dep. Elect. Eng., Univ. Queensland, Tech. Rep., Apr. 1992.

[5] D. G. Stork. S. Walker. M. Burns. B. Jackson. Preadaptation in neural circuits. *Proc. Int. Joint Conf. "Neural Networks"*, vol. I, Washington, DC, 1990, pp. 202–205.

[6] D. White. P. Ligomenides. GANNet: A genetic algorithm for optimizing topology and weights in neural network design. *Proc. Int. Workshop "Artificial Neural Networks (IWANN'93)"*, Lecture Notes in Computer Science, vol. 686. Berlin, Germany: Springer-Verlag, 1993, pp. 322–327.

[7] Y.Liu. X. Yao. Evolutionary design of artificial neural networks with different nodes. *Proc. 1996 IEEE Int. Conf. "Evolutionary Computation (ICEC'96)"*, Nagoya, Japan, pp. 670–675.

[8] A. Abraham. Meta learning evolutionary artificial neural networks. *Neurocomputing*, vol. (56) 1-38, 2004.

[9] F. H. F. Leung, H. K. Lam, S. H. Ling, P. K. S. Tam. Tuning of the structure and parameters of a neural network using an improved genetic algorithm. *IEEE Transactions on Neural Neworks*. Vol. 14 (1) 79- 88, 2003.

[10] P. Palmes, S. Usui. Robustness, Evolvability and Optimality in Evolutionary Neural Networks". *Biosystems*, vol. 82 (2) 168-188, 2005.

[11] A. V. Sebald, K. Chellapilla. On making problems evolutionarily friendly, part I: Evolving the most convenient representations. In V. W. Porto. N.Saravanan, D. Waagen. A. E. Eiben. (Eds.) *Evolutionary Programming VII: Proc. 7th Annu Conf. "Evolutionary Programming"*, vol. 1447 of Lecture Notes in Computer Science, Berlin, Germany: Springer-Verlag, 1998, pp. 271–280.

[12] R. Miikkulainen. Evolving neural networks. In *Proceedings of the 2007 GECCO Conference Companion on Genetic and Evolutionary Computation* GECCO '07. ACM, New York, NY, 2007, pp. 3415-3434.

[13] F. J. Gomez. R. Miikkulainen. Active guidance for a finless rocket through neuroevolution. *Proceedings of the Conference "Genetic and Evolutionary Computation (GECCO-2003)"*. Berlin: Springer Verlag. 2003.

[14] C. Igel. Neuroevolution for reinforcement learning using evolution strategies. In R. Sarker R.Reynolds. H. Abbass. K. C. Tan. B. McKay. D. Essam. T.Gedeon (eds.) "*Congress on Evolutionary Computation 2003 (CEC 2003)"* Piscataway, NJ: IEEE Press. 2003. pp. 2588–2595.

[15] E. Volna. Forming neural network design through evolution".. In K. Madani (ed.). *Proceedings of the 3th International Workshop on "Artificial Neural Networks and Intelligent Information Processing (*ANNIIP 2007)". In conjunction with ICINCO 2007. Angers, France 2007, pp. 13-20.

[16] Volná, E. "Learning algorithm which learns both architectures and weights of feedforward neural networks". *Neural Network World. Int. Journal on Neural & Mass-Parallel Comp. and Inf. Systems*. 8 (6): 653-664, 1998.

[17] S. Garfinger. *PGP: Pretty Good Privacy*. Computer Press, Praha 1998.

[18] E. Cantu-Paz. Adaptive sampling for noisy problems. In *Genetic and Evolutionary Computation Conference*, pages 947--958, Springer 2004.

[19] P. A., Castillo, J.J. Merelo, M. G. Arenas, and G. Romero. Comparing evolutionary hybrid systems for design and optimization of multilayer perceptron structure along training parameters. In *Information Sciences*, Vol 177 (14) 2884-2905, 2007.

***Eva Volna** graduated at the Slovak Technical University in Bratislava and defended PhD. thesis with title "Modular Neural Networks".*

She has been working as an assistant professor at the Department of Computer Science, University of Ostrava (Czech Republic) from 1992.

Her interests include artificial intelligence, artificial neural networks, evolutionary algorithms, and cognitive science.

She is author of 37 publications - 9 articles in reviewed journals, 28 articles in proceedings of international and national conferences, 8 teaching texts and 3 graduation theses.

computing@computingonline.net
www.computingonline.net

ISSN 1727-6209
International Scientific
Journal of Computing

DIRECT AND INDIRECT CLASSIFICATION OF HIGH FREQUENCY LNA GAIN PERFORMANCE – A COMPARISON BETWEEN SVMS AND MLPS

Peter C. Hung, Seán F. McLoone, Ronan Farrell

Institute of Microelectronics and Wireless Systems, Department of Electronic Engineering, National University of Ireland Maynooth, Maynooth, Co. Kildare, Ireland,
{phung, sean.mcloone, rfarrell}@eeng.nuim.ie, http://imws.eeng.nuim.ie/

Abstract: *The task of determining low noise amplifier (LNA) high-frequency performance in functional testing is as challenging as designing the circuit itself due to the difficulties associated with bringing high frequency signals off-chip. One possible strategy for circumventing these difficulties is to inferentially estimate the high frequency performance measures from measurements taken at lower, more accessible, frequencies. This paper investigates the effectiveness of this strategy for classifying the high frequency gain of the amplifier, a key LNA performance parameter. An indirect Multilayer Perceptron (MLP) and direct support vector machine (SVM) classification strategy are considered. Extensive Monte-Carlo simulations show promising results with both methods, with the indirect MLP classifiers marginally outperforming SVMs.*

Keywords: *LNA, Functional testing, Classification, Support Vector Machines, Multilayer Perceptrons.*

1. INTRODUCTION

Functional testing of radio frequency integrated circuits (RFIC) is becoming an increasingly difficult problem for IC manufacturers, especially for RF components that operate in the multi-gigahertz range. Two main problems exist when working at these frequencies: relaying the high-frequency signals to and from the automatic test equipment (ATE) without affecting the performance of the circuits under test; building RF production testers that are not prohibitively expensive. While advances in technology and market requirements have seen rapid growth in high-frequency and high integration RFIC designs, testing practice has not followed suit. Indeed, reliable high-frequency testing has become the dominant factor in the cost and time-to-market of novel wireless products [1]. Consequently, developing cost-efficient testing solutions is becoming an increasingly important research topic [2-4].

Some of the proposed schemes for RFIC testing are based on an end-to-end strategy in which the output of the transmitter and the input of the receiver are linked through a loop-back connection. In this configuration, the testing of the complete system is carried out without any external stimulus by employing the on-chip digital hardware available. Unfortunately, this solution is not always applicable to all kinds of RF components. Other recent proposals for RF system testing have focused on the development of methodologies and algorithms for automated test, and Design for Testability (DfT). One approach, Built-In-Test (BIT) uses additional circuitry that allows high frequency tests to be performed on-chip and then evaluated using lower frequency or DC external testers [3-5]. An example of this approach by Bhattacharya and Chatterjee used a CMOS RMS detector for testing wireless transceivers [6, 7]. By employing a three-stage rectifier, they converted the high frequency circuit output into a DC signal which was then correlated with the equivalent test specification values of the circuit-under-test (CUT). In this manner, the actual performance of the system, under certain conditions, could be estimated. However, when considering BIT testing, issues such as the on-chip area and power consumption required for the additional embedded circuitry can add significantly to the cost of the design.

In this paper a different approach is considered. As many RFICs show strong correlation between their responses to circuit parameter variation at different frequencies, it is hypothesised that knowledge of the responses at lower frequencies may provide sufficient information to allow classification of responses at higher frequencies.

To investigate this hypothesis, the testing requirement of a low noise amplifier (LNA) is used as a case study. LNAs are key components in modern wireless systems as they provide the initial amplification for weak signals received from antennae. For our study, a standard LNA design utilising United Microelectronics Corporation's (UMC) 0.18 μm silicon process technology and designed to operate at 2.4 GHz was selected [8]. The LNA circuit consisted of 2 bias transistors (0.18 μm channel length), 4 RF transistors (0.5 μm channel width), 4 resistors, 3 capacitors and 4 inductors and was deemed to be functioning correctly if the value of S21 @ 2.4 GHz was in the range 14.7 dB to 17.2 dB and faulty otherwise. S21 is the forward transmission coefficient or forward voltage gain (hereafter referred to as 'gain') of the amplifier and is a critical RF circuit performance measure. The '@' notation is used to indicate the circuit excitation frequency at which the measurement is taken.

Random circuit parameter perturbations representative of typical manufacturing process variations were generated and the value of S21 recorded at different frequencies. Fig.1 shows a plot of the correlation that exists between variations in gain (S21 @ 2.4 GHz) and variations in the same parameter computed at other frequencies. There is a strong, but decreasing, correlation evident as the circuit excitation frequency moves away from the operating frequency.

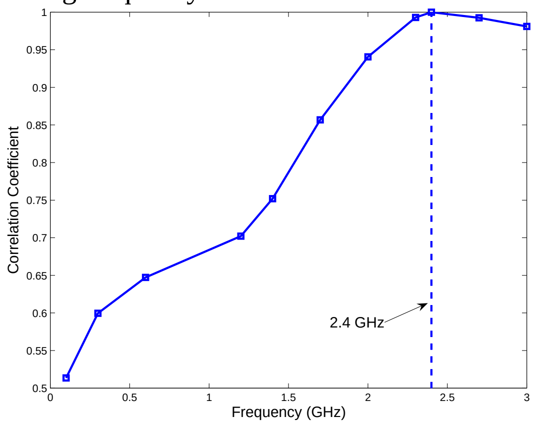


Fig. 1 – Correlation between variations in S_{21} computed at different frequencies and the variations in S_{21} computed at the target frequency (2.4 GHz).

Even when the correlation is high, circuit performance classification on the basis of these lower frequencies is not straightforward, as demonstrated in Fig.2. This shows the relationship between S21 @ 2.4 GHz and the values computed at: (a) 2.0 GHz and (b) 0.1 GHz, respectively, and highlights the fact that even when the correlation is greater than 90% it is not possible to discriminate between the 'good' and 'bad' circuits effectively. In fact, simple thresholding on the basis of S21 @ 2.0 GHz leads to a misclassification rate of greater than 20%. The misclassification rate increases rapidly as the frequency is reduced and reaches 43.6% for S21 @ 0.1 GHz. The rapid deterioration in performance is a result of the localised influence of some parameter variations and the complex nonlinear interaction between circuit components.

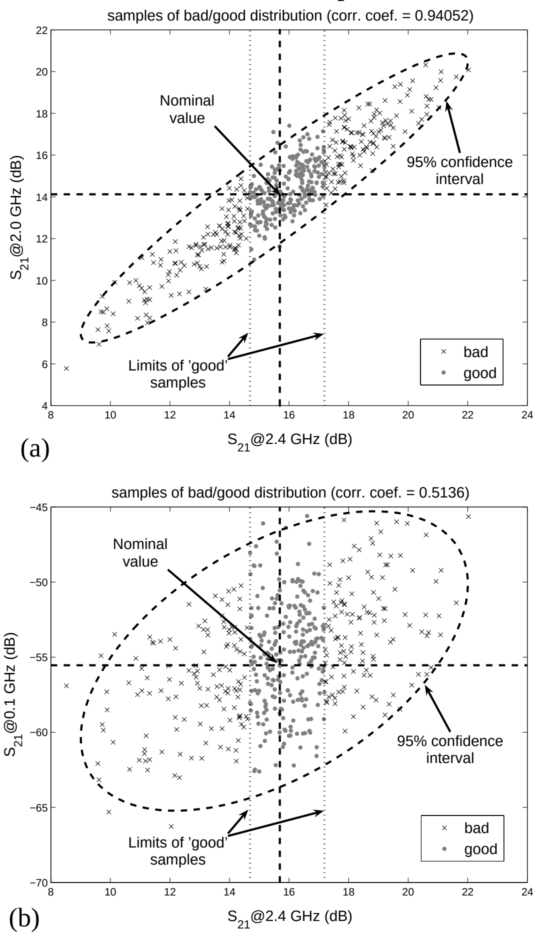


Fig. 2 – S_{21} parameter relationships for the 2.4 GHz LNA model used in circuit simulations: (a) S_{21} @ 2.0 GHz and (b) S_{21} @ 0.1 GHz plotted against S_{21} at the operating frequency.

Since the shape of the frequency response of an LNA is a deterministic nonlinear function of its component parameters, better classification performance can be expected if the information from several low-frequency measurements can be combined. To that end, this paper considers the possibility of classifying circuit performance using machine learning techniques. Two strategies are investigated. In the first, an Artificial Neural Network (ANN) is trained to predict the value of S_{21} @ 2.4 GHz from the values measured at other frequencies and then a thresholding rule is applied to this prediction to perform the circuit classification, while in the second, a Support Vector Machine (SVM) is trained to directly classify circuit performance on the basis of the low-frequency S_{21} measurements. These two machine learning techniques and the LNA classification methodology are introduced in Section 2. The simulation study is then described in Section 3 followed by the results in Section 4. Finally, the conclusions of the study are presented in Section 5.

2. MACHINE LEARNING LNA PERFORMANCE CLASSIFICATION

Defining the set of N low-frequency S_{21} measurements of the i^{th} LNA circuit as the feature vector $\mathbf{x}_i$ (row vector) and the corresponding class label y_i, with $y_i = +1$ indicating 'good' and $y_i = -1$ indicating 'bad', we can generate a set of L training data examples,

$$(\mathbf{X},\mathbf{y}) = (\mathbf{x}_1, y_1), \ldots (\mathbf{x}_i, y_i), \ldots (\mathbf{x}_L, y_L) \in \Re^N, \quad (1)$$

with which to train an LNA circuit classifier to estimate a decision function $f(\mathbf{x})$

$$f(\mathbf{x}) : \Re^N \to \{\pm 1\}, \quad (2)$$

that can then be used to classify new circuits. Here, 'good' and 'bad' are determined by a threshold function, z_{th} applied to S_{21} @ 2.4 GHz, that is

$$z_{\text{th}}(x) = \begin{cases} +1 & \text{if} \quad 14.7 < x < 17.2 \\ -1 & \text{otherwise} \end{cases} \quad (3)$$

The decision function in (2) can be estimated directly from the training data by using, for example, a SVM classifier. Alternatively it can be estimated indirectly by first predicting the value of S_{21} @ 2.4 GHz from the feature vector,

$$g(\mathbf{x}) \to S_{21}\,@\,2.4\text{GHz}, \quad (4)$$

and then using a threshold function to perform the classification, that is

$$f(\mathbf{x}) : z_{\text{th}}(g(\mathbf{x})) \to \{\pm 1\}, \quad (5)$$

The difference between the two implementations is illustrated graphically in Fig.3.

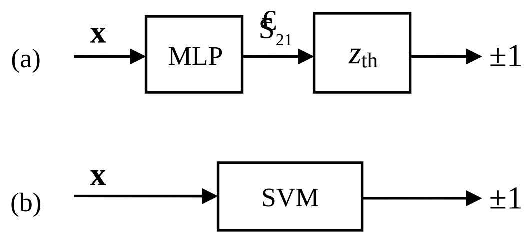


Fig. 3 – Two classifier implementations: (a) indirect MLP; and (b) direct SVM.

An artificial neural network such as a Multilayer Perceptron (MLP) can be used to learn the nonlinear mapping represented by (4).

2.1 SUPPORT VECTOR MACHINES

SVMs, first proposed by Vladimir Vapnik in 1963 [9], are a supervised linear learning technique widely used for classification problems. They are known to perform binary classification well in many practical applications. Consider a separating hyperplane that divides two classes of data:

$$\mathbf{w} \cdot \mathbf{x} - b = 0, \quad \mathbf{w} \in \Re^N, b \in \Re, \quad (6)$$

where $\mathbf{w}$ and b are unknown coefficients, and two additional hyperplanes that are parallel to the separating hyperplane:

$$\begin{aligned} \mathbf{w} \cdot \mathbf{x} - b &= 1 \\ \mathbf{w} \cdot \mathbf{x} - b &= -1 \end{aligned} \quad (7)$$

Defining the margin as the perpendicular distance between the parallel hyperplanes, the optimal hyperplane is the one which results in the maximum margin of separation between the two classes.

Mathematically the problem can be expressed as

$$\max_{\mathbf{w}} \frac{2}{|\mathbf{w}|} \equiv \min_{\mathbf{w}} (\mathbf{w} \cdot \mathbf{w}), \quad (8)$$

subject to

$$y_i(\mathbf{w} \cdot \mathbf{x}_i - b) \geq 1, \quad i = 1,2,\ldots,L. \quad (9)$$

This is a constrained quadratic optimisation problem whose solution w has an expansion [10]

$$\mathbf{w} = \sum_i v_i \mathbf{x}_i \quad (10)$$

where $\mathbf{x}_i$ are the subset of the training data, referred to as support vectors, located on the parallel hyperplanes, and v_i are the corresponding weighting factors. The linear SVM (LSVM) decision function is then given by

$$f_{\text{LSVM}}(\mathbf{x}) = z_{\text{SVM}}(\mathbf{w} \cdot \mathbf{x} - b), \quad (11)$$

where z_{SVM} is defined as

$$z_{\text{SVM}}(\phi) = \begin{cases} +1 & \text{if } \phi \geq 0 \\ -1 & \text{if } \phi < 0 \end{cases} \quad (12)$$

The decision function in (11) can be rewritten as

$$f_{\text{LSVM}}(\mathbf{x}) = z_{\text{SVM}}\left(\sum_i v_i(\mathbf{x}\cdot\mathbf{x}_i) - b\right), \tag{13}$$

with the result that it is only dependent on dot products between the test data vector, $\mathbf{x}$, and the support vectors. This important property allows SVMs to be extended to problems where nonlinear partitions of data sets are required. This is achieved by replacing the dot products by a kernel function $k(.)$ which meets the Mercer's condition [11]:

$$k(\mathbf{x}_i,\mathbf{x}_j) = \Phi(\mathbf{x}_i)\cdot\Phi(\mathbf{x}_j). \tag{14}$$

This maps the data into a higher dimension feature space where linear SVM classification can be performed. Note the resulting decision function, in the original data space, will be nonlinear and takes the form

$$f_{\text{SVM}}(\mathbf{x}) = z_{\text{SVM}}\left(\sum_i^L v_i.k(\mathbf{x},\mathbf{x}_i) - b\right). \tag{15}$$

In non-separable problems where different classes of data overlap, slack variables can be introduced so that a certain amount of training error or data residing within the margin is permitted. This gives rise to a 'soft margin' optimisation function [11, 12]. To give users the ability to adjust the amount of training error allowed in the optimisation, a smoothing parameter C is incorporated into the soft margin function, with a larger C corresponding to assigning a larger penalty to errors.

The Gaussian radial basis function (RBF), defined as

$$k_{\text{RBF}}(\mathbf{x}_i,\mathbf{x}_j) = \exp\left(-\frac{\left|\mathbf{x}_i-\mathbf{x}_j\right|^2}{2\sigma^2}\right) \tag{16}$$

is a popular choice of SVM kernel and the one selected for this application. The parameter, σ, controls the width of the kernel and is determined as part of the classifier training process.

2.2 ARTIFICIAL NEURAL NETWORKS

Neural networks [13] are one of the best known and most commonly-used machine learning techniques. There are various configurations and structures of NNs, but all contain an array of neurons that are linked together, usually in multiple layers. In this application a single hidden layer Multilayer Perceptron (MLP) topology is chosen because of its universal function approximation capabilities, good generalisation properties and the availability of robust efficient training algorithms [14].

The output of a single hidden layer MLP can be written as a linear combination of sigmoid functions (i.e. neurons),

$$g(\mathbf{x},\mathbf{w}_{\text{NN}}) = b^h + \sum_i w_i^h sig_i(\mathbf{x}), \tag{17}$$

where

$$sig_i(\mathbf{x}) = \frac{1}{1+\exp(\mathbf{w}_i^u\cdot\mathbf{x}+b_i^u)}. \tag{18}$$

Here, w_i^h, $\mathbf{w}_i^u$, b_i^u ,($i=1,2,\cdots,M$) and b^h are weights and biases which collectively form the network weights vector, $\mathbf{w}_{\text{NN}}$. Defining a Mean Squared Error (MSE) cost function over the training data

$$E(\mathbf{w}_{\text{NN}}) = \frac{1}{L}\sum_{p=1}^{L}(g(\mathbf{x}_p,\mathbf{w}_{\text{NN}})-d_p)^2, \tag{19}$$

with d_p corresponding to the desired network output for the p^{th} training pattern (i.e. S_{21} @ 2.4 GHz), the optimum weights can be determined using gradient based optimisation techniques.

3. SIMULATION STUDY

Circuit simulations were required to evaluate the potential for employing multiple low-frequency S_{21} measurements to classify LNA S_{21} performance at high frequency, and to compare the performance of the proposed machine learning classifiers. Hence, a Monte Carlo simulation study was undertaken using a 2.4 GHz LNA model implemented in Eldo® [8]. Uniform random variations were introduced into 38 of the model parameters and 10,000 circuit simulations were performed. These parameter variations were chosen to emulate typical industrial LNA manufacturing process variations, such as the lithographic dimensions that affect component resistance and capacitance. While in practice circuit parameters might be expected to vary normally around their nominal values, uniform distributions were chosen to give an even coverage of the LNA parameter space. Catastrophic failures, such as short-circuits, were not considered as these can be identified relatively easily using existing IC testing techniques.

For each circuit the S_{21} performance parameter was recorded at 0.1, 0.3, 0.6, 1.2, 1.4, 1.7 and 2.0 GHz and also at the operating frequency (2.4 GHz). This data was then normalised to have zero mean and unit variance and divided into training and test data sets, each containing 5,000 samples.

As the study focused on the viability of performance classification using low frequency S_{21} measurements, two different feature vectors were considered in the study, containing measurements from 0.1 to 1.4 GHz and 0.1 to 2.0 GHz, respectively, that is:

$$\mathbf{x}_{1.4} = [S_{21}^{0.1}, S_{21}^{0.3}, S_{21}^{0.6}, S_{21}^{1.2}, S_{21}^{1.4}], \qquad (20)$$

and

$$\mathbf{x}_{2.0} = [S_{21}^{0.1}, S_{21}^{0.3}, S_{21}^{0.6}, S_{21}^{1.2}, S_{21}^{1.4}, S_{21}^{1.7}, S_{21}^{2.0}]. \qquad (21)$$

Here, S_{21}^{f} denotes the value of S_{21} at f GHz. In each case the target MLP model output is $S_{21}^{2.4}$ while the target labels for the SVM classifier are given by $z_{\text{th}}(S_{21}^{2.4})$.

3.1 MLP TRAINING

MLP training was performed using the hybrid BFGS training algorithm [14] with stopped minimisation used to prevent over-fitting [15]. The optimum number of neurons (M) was determined for each model by systematically evaluating different network sizes and selecting the network with the minimum MSE on the test data set. Training was repeated ten times for each network size to allow for random weight initialisations and the best set of weights recorded in each case.

The optimum network sizes and resulting model fit, measured in terms of the correlation with the true value of $S_{21}^{2.4}$, are summarised in Table 1. For comparison purposes, the correlation between $S_{21}^{2.4}$ and the single frequency measurements at 1.4 and 2.0 GHz are also given.

Table 1. Optimum MLP classifier model dimensions and resulting model fit (correlation coefficient)

Feature Vector	Network Dimensions (input, hidden, output)	Model Fit
$\mathbf{x}_{1.4}$	MLP(5,12,1)	0.9364
$\mathbf{x}_{2.0}$	MLP(7,15,1)	0.9979
$S_{21}^{1.4}$	-	0.7537
$S_{21}^{2.0}$	-	0.9408

As expected, the exploitation of multiple frequencies results in much better predictability of $S_{21}^{2.4}$ than using the measurement at a single frequency. Notably, the information provided by $S_{21}^{2.0}$ is still marginally greater than the combined information provided by all measurements up to 1.4 GHz. The classification performance of these networks, when employed in the indirect LNA classifier scheme, will be reported in Section 4.

3.2 SVM TRAINING

SVM training was performed using two Matlab® packages: LibSVM v2.84 [16] and simpleSVM [17]. The kernel width parameter σ and smoothing parameter C were fine tuned in LibSVM using a five-fold cross validation strategy.

Initial SVM results were quite poor despite expectations of superior performance to indirect classification using MLPs (results presented in Section 4). It was determined that this might be due to the bimodal distribution of the out-of-specification circuits forming the 'bad' class, i.e. it consists of two segments separated by the 'good' class, as shown in Fig.4.

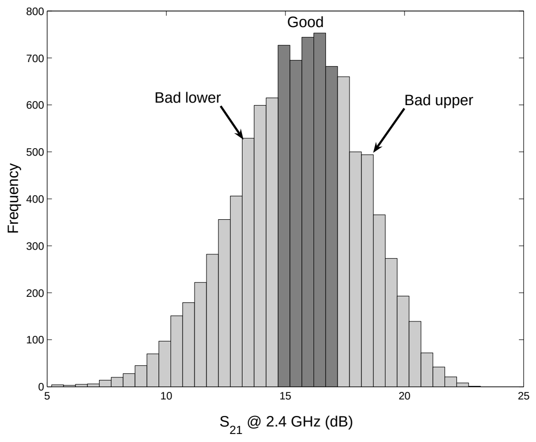


Fig. 4 – Histogram of 'good' and 'bad' circuits as defined by the S_{21} performance criteria.

The *a priori* knowledge of the distribution of the 'bad' circuits can be taken into account by splitting the 'bad' samples into 'bad lower' and 'bad upper' samples, thereby introducing 3 classes – 'bad lower', 'good' and 'bad upper'. SVM classification is then performed in two-stages as shown in Fig.5. Firstly, two binary SVMs are trained, one to classify LNAs as either 'bad lower' or 'not bad lower', and one to classify LNAs as either 'bad upper' or 'not bad upper'. Then the overall classification is obtained from a NOR logic combination of the individual decision functions, as shown in Table 2.

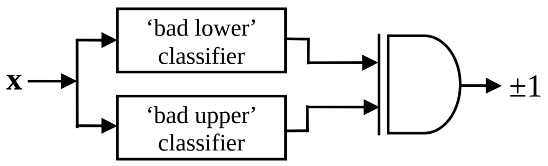


Fig. 5 – SVM classification for 3 classes.

This three-class SVM approach is denoted SVM3 while the original two-class SVM classifier will be referred to as SVM2.

Table 2. Logic table of two-stage SVM3 classification

Stage 1	Stage 2	Final decision
'bad lower' (S_{21} < 14.2 dB)?	**'bad upper' (S_{21} > 17.2 dB)?**	**Circuit 'good'/'bad'?**
no	no	'good'
yes	no	'bad'
no	yes	'bad'
yes	yes	assumed 'bad'*

* Though theoretically unattainable, in practice such situations arise due to incorrect SVM predictions.

4. RESULTS

The performance of the MLP, SVM2 and SVM3 LNA classifiers was measured in terms of the following metrics computed on the test data set:

- **GPR**: good pass rate – Percentage of 'good' LNAs correctly classified as 'good'
- **BFR**: bad fail rate – Percentage of 'bad' LNAs correctly classified as 'bad'
- **FR**: failure rate – Percentage of LNAs classified as 'good' that are in fact 'bad'
- **MCR**: misclassification rate – Percentage of LNAs incorrectly classified

To provide robust estimates, the metrics were computed by averaging over 100 batches of LNAs generated from the test data set using sampling with replacement. Each batch consisted of 500 'good' and 500 'bad' circuits randomly selected (with replacement) from a total of 1,795 'good' and 3,205 'bad' examples in the test pool.

Since the good pass rate (GPR) and bad fail rate (BFR) of a classifier vary as a function of the classification threshold, with one increasing as the other decreases, the threshold can be adjusted to control one or other of these metrics. This is achieved by adding an offset, α , to the decision function applied to the classifier variable. In the case of SVMs, for example, the decision function $z_{\text{SVM}}(\phi)$ in (12) becomes $z_{\text{SVM}}(\phi-\alpha)$.

Tables 3 and 4 show the mean performance of the classifiers at 90% and 75% BFR, respectively. Here, the classifier offset was adjusted to achieve the target BFRs. This approach was adopted as it reflects the importance in the electronics industry of controlling the number of faulty components released to the market. The result for classification on the basis of single frequency measurements at 1.4 and 2.0 GHz are also included for comparison.

Table 3. Mean performance of the MLP and SVM LNA classifiers at 90% BFR

Feature Vector	Method	GPR (%)	FR (%)	MCR (%)
$\mathbf{x}_{1.4}$	MLP	57.11	14.88	26.45
	SVM2S*	46.61	17.61	31.69
	SVM2	53.48	15.72	28.26
	SVM3S*	46.87	17.55	31.56
	SVM3	54.26	15.53	27.87
$\mathbf{x}_{2.0}$	MLP	99.70	9.09	5.15
	SVM2S*	92.49	9.74	8.75
	SVM2	94.29	9.57	7.85
	SVM3S*	92.32	9.75	8.84
	SVM3	94.10	9.59	7.95
$S_{21}^{1.4}$	-	19.98	33.29	45.01
$S_{21}^{2.0}$	-	55.48	15.24	27.27

* From simpleSVM toolbox.

Table 4. Mean performance of the MLP and SVM LNA classifiers at 75% BFR

Feature Vector	Method	GPR (%)	FR (%)	MCR (%)
$\mathbf{x}_{1.4}$	MLP	81.03	23.55	21.99
	SVM2S*	77.40	24.39	23.80
	SVM2	81.74	23.40	21.63
	SVM3S*	78.44	24.15	23.28
	SVM3	82.12	23.32	21.44
$\mathbf{x}_{2.0}$	MLP	100.00	19.97	12.50
	SVM2S*	98.81	20.17	13.10
	SVM2	99.13	20.13	12.94
	SVM3S*	98.79	20.17	13.10
	SVM3	99.12	20.13	12.94
$S_{21}^{1.4}$	-	47.13	34.63	38.94
$S_{21}^{2.0}$	-	84.65	22.78	20.17

* From simpleSVM toolbox.

Comparing the different implementations of SVM, it is found that simpleSVM's performance is slightly inferior to LibSVM. This is further highlighted in Fig.6, which shows the operating curves of the classifiers obtained with each SVM package for the SVM3 approach. While the difference is marginal when using the $\mathbf{x}_{2.0}$ feature vector, it is significant when using $\mathbf{x}_{1.4}$, which corresponds to the more demanding of the classification problems. It is not clear why this should be the case, but it is believed that the simpleSVM solution may not be precise when there is a large degree of overlap between classes.

It is noted that high GPRs are always accompanied by correspondingly low values of FR and MCR. As expected, the performance of all classifiers deteriorates when only the lower frequency S_{21} measurements ($\mathbf{x}_{1.4}$) are considered, though SVM3 and MLP still provide comparable performance to classification using a single S_{21} measurement at 2.0 GHz.

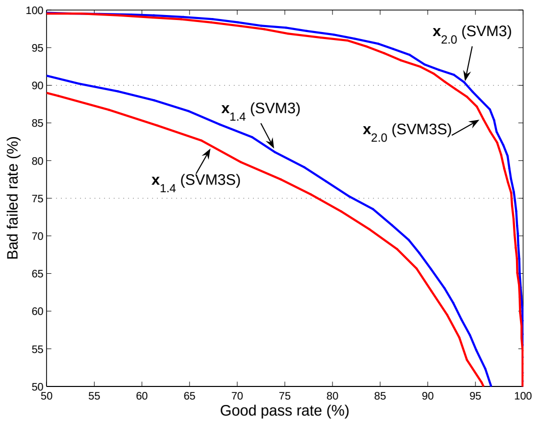


Fig. 6 – Operating curves (BFR vs. GPR) for $x_{1.4}$ and $x_{2.0}$ obtained with simpleSVM (SVM3S) and LibSVM (SVM3).

When comparing the MLP and SVM3 classifiers, it can be seen that the MLP gives the best results at almost all BFRs, as shown in Fig.7. When using $\mathbf{x}_{1.4}$ SVM3 is only marginally inferior to the MLP. However, the GPR gap increases to more than 5% when the higher frequency feature vector, $\mathbf{x}_{2.0}$, is considered at 90% BFR.

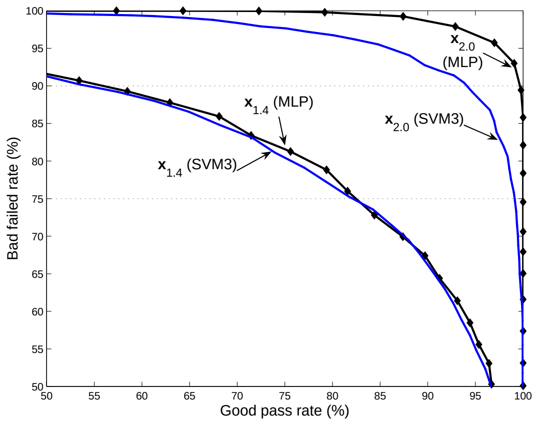


Fig. 7 – Operating curves (BFR vs. GPR) for MLP and SVM3 classifiers (for $x_{1.4}$ and $x_{2.0}$).

Interestingly, the direct SVM classifiers are not able to outperform the indirect MLP classifiers, even when the *a priori* knowledge of the bimodal distribution of the out-of-specification LNAs is taken into account. Indeed, the SVM2 and SVM3 classifiers give almost identical results indicating that SVMs can comfortably handle complex data distributions and that, therefore, exploitation of this knowledge is unnecessary. Consequently, it can be concluded that while, in general, SVMs may be the natural setting for classification tasks, the indirect classification approach can, in some instances, achieve better results. This may be a reflection of the fact that the indirect approach allows exploitation of more detailed information about the underlying mapping in its learning process.

In a manufacturing context the operating curves are a useful tool for assessing the trade-offs that can be obtained with each classifier. For example, for a target BFR of 90% a GPR of 55% is obtained with $\mathbf{x}_{1.4}$ as a feature vector. This increases to over 99% with $\mathbf{x}_{2.0}$. Alternatively, a 100% BFR can be obtained when using $\mathbf{x}_{2.0}$ if the GPR is dropped to 75%. The level of wastage at $\mathbf{x}_{1.4}$ may well be acceptable to manufacturers when the costs of generating the additional test signals needed for $\mathbf{x}_{2.0}$ are taken into account.

5. CONCLUSIONS

Functional testing of high-frequency LNAs is becoming a prohibitively expensive and time-consuming exercise, due to the difficulties with bringing such signals off-chip. This paper has investigated a novel testing strategy in which machine learning classifiers are used to predict high-frequency LNA gain performance by combining information from several lower frequency measurements. Results have been presented for both direct SVM and indirect MLP classifier implementations. These show that the proposed strategy has the potential to significantly extend the operating frequency range of existing ATE. For example, in the gain classification case study considered, the operating frequency range was extended by 20% at 2 GHz and 42% at 1.4 GHz.

Of the two classifier implementations considered the indirect MLP classifier yielded marginally superior performance and is the preferred approach for this application.

ACKNOWLEDGEMENTS

Research presented in this paper was funded by Enterprise Ireland Commercialisation Fund (EI CFTD/2003/304) under the National Development Plan. The authors gratefully acknowledge this support.

6. REFERENCES

[1] J. Ferrario, R. Wolf, H. Ding, Moving from mixed signal to RF test hardware development. *IEEE Int. Test Conference* (2001) pp. 948–956.

[2] W. Y. Lau, Measurement challenges for on-wafer RF-SOC test. *27th Annual IEEE/SEMI Int. Elect. Manufact. Tech. Symp.* (2002) pp. 353–359.

[3] M. Negreiros, L. Carro, A. Susin, Low cost on-line testing of RF circuits. *10th IEEE Int. On-Line Testing Symp.* (2004) pp. 73–78.

[4] D. C. Doskocil, Advanced RF built in test. *AUTOTESTCON '92 IEEE Sys. Readiness Tech. Conf.* (1992) pp. 213–217.

[5] M. E. Goff, C. A. Barratt, DC to 40 GHz MMIC power sensor. *Gallium Arsenide IC Symp.* (1990) pp. 105–108.

[6] S. Bhattacharya, A. Chatterjee, Use of embedded sensors for built-in-test RF circuits, *IEEE Int. Test Conf.* (2004) pp. 801 – 809.

[7] S. S. Akbay, A. Chatterjee, Feature extraction based built-in alternate test of RF components using a noise reference, *22nd IEEE VLSI Test Symp.* (2004) pp. 273 – 278.

[8] Mentor Graphics Corporation, *Eldo User Manual* (2005).

[9] V. Vapnik, A. Lerner, Pattern recognition using generalised portrait method. *Automation and Remote Control* 24 (1963).

[10] M.A. Hearst, SVMs – a practical consequence of learning theory. *IEEE Intelligent Sys.* (1998) pp. 18–21.

[11] C. J. Burges, A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery* 2 (1998) pp. 121–167.

[12] C. Cortes, V. Vapnik, Support vector networks. *Machine Learning* 20 (1995) pp. 273–297

[13] S. Haykin, *Neural Networks: A comprehensive foundation.* 2nd edn. Prentice Hall, New Jersey (1998).

[14] S. McLoone, M. Brown, G. Irwin, G. Lightbody, A hybrid linear/nonlinear training algorithm for feedforward neural networks. *IEEE Trans. on Neural Networks* 9 (1998) pp. 669-684.

[15] J. Sjöberg, L. Ljung, Overtraining, regularization, and searching for minimum with application to neural networks. *Int. J. Control* 62 (1995) pp. 1391-1407.

[16] C.-C. Chang, C.-J. Lin, LibSVM: a library for support vector machines (2001). Software available at http://www.csie.ntu.edu.tw/~cjlin/-libsvm.

[17] S. V. N. Vishwanathan, A. J. Smola, M. N. Murty, SimpleSVM. *Proc. 20th Int. Conf. Machine Learning* (2003).

Peter C. Hung *obtained his M.Eng. in Electrical and Electronic Engineering at Queen's University Belfast, Northern Ireland in 2001. He graduated with a Ph.D. in Electrical and Electronic Engineering from the Virtual Engineering Centre, QUB in 2005. His thesis was on thermocouple sensor fusion and characterisation.*

Currently he is a post-doctoral research fellow in the National University of Ireland, Maynooth, Ireland.

His research interests include system identification, online and offline signal processing, fault diagnosis, pattern classification, and machine learning. His current projects are in the research fields of fast temperature measurement and integrated circuit testing.

Seán F. McLoone *received the M.Eng. degree in electrical and electronic engineering and the Ph.D. degree in control engineering from Queen’s University Belfast, Belfast, U.K., in 1992 and 1996, respectively.*

He is a Senior Lecturer with the Department of Electronic Engineering, National University of Ireland Maynooth, Maynooth, Ireland.

His research interests are in the general area of data-based modeling and analysis of dynamical systems. This encompasses techniques ranging from classical system identification, fault diagnosis, and statistical process control to modern artificial-intelligence-inspired adaptive learning algorithms and optimization techniques. His research has a strong application focus, with many projects undertaken in collaboration with industry in areas such as process monitoring, control and optimization, time series prediction, and in-line sensor characterization.

Ronan Farrell *graduated from University College Dublin in 1993 with a B.E and returned later to receive his Ph.D. degree in 1998. After receiving his Ph.D. he joined Parthus Technologies as a mixed signal integrated circuit designer. In 2001, he left Parthus to join NUI Maynooth as a lecturer in the Department of Electronic Engineering. In 2004, he became an SFI theme leader for high frequency electronics research within the Centre for Telecommunications Value-Chain Driven Research (CTVR) which is the academic partner to Bell Labs Ireland. This Centre, with Bell Labs, remains the largest single investment (€79 million) by any research funding agency in Ireland. As of December 2005, Ronan was appointed director of the Institute of Microelectronics and Wireless Systems at NUI Maynooth, a multidisciplinary centre focused on applied research in wireless systems and their enabling technologies.*

Ronan’s personal research interests include circuit design, microelectronics, and wireless technologies.

computing@computingonline.net
www.computingonline.net

ISSN 1727-6209
International Scientific Journal of Computing

DATA DIMENSIONALITY REDUCTION FOR NEURAL BASED CLASSIFICATION OF OPTICAL SURFACES DEFECTS

Matthieu Voiry [1&2], Kurosh Madani [1], Véronique Amarger [1], Joël Bernier [2]

[1] Image, Signal and Intelligent Systems Laboratory (LISSI / EA 3956), Senart Institute of Technology, University PARIS XII, Av. Pierre Point, F-77127 Lieusaint, France,
{voiry ; madani ; amarger}@univ-paris12.fr, http://www.univ-paris12.fr/

[2] SAGEM REOSC
Avenue de la Tour Maury, Saint Pierre du Perray, 91280, France
{mathieu.voiry or joel.bernier}@sagem.com

Abstract: *A major step for high-quality optical surfaces faults diagnosis concerns scratches and digs defects characterization in products. This challenging operation is very important since it is directly linked with the produced optical component's quality. A classification phase is mandatory to complete optical devices diagnosis since a number of correctable defects are usually present beside the potential "abiding" ones. Unfortunately relevant data extracted from raw image during defects detection phase are high dimensional. This can have harmful effect on the behaviors of artificial neural networks which are suitable to perform such a challenging classification. Reducing data dimension to a smaller value can decrease the problems related to high dimensionality. In this paper we compare different techniques which permit dimensionality reduction and evaluate their impact on classification tasks performances.*

Keywords: *Computer Aided Diagnosis Systems (CADS), Artificial Intelligent systems, Industrial applications, Artificial Neural Network, Dimensionality Reduction, Curvilinear Component Analysis (CCA), Curvilinear Distance Analysis (CDA), Self Organizing Maps (SOM).*

1. INTRODUCTION

We are involved in fault diagnosis of optical devices in industrial environment. In fact, classification of detected faults is among chief phases for succeeding in such diagnosis. Aesthetic flaws, shaped during different manufacturing steps, could provoke harmful effects on optical devices' functional specificities, as well as on their optical performances by generating undesirable scatter light, which could seriously degrade the expected optical features. Taking into account the above-mentioned points, a reliable diagnosis of these defects in high-quality optical devices becomes a crucial task to ensure products' nominal specification and to enhance the production quality. Moreover, the diagnosis of these defects is strongly motivated by manufacturing process correction requirements in order to guarantee mass production (repetitive) quality with the aim of maintaining acceptable production yield.

Unfortunately, detecting and measuring such defects is still a challenging dilemma in production conditions and the few available automatic control solutions remain ineffective. That's why, in most of cases, the diagnosis is performed on the basis of a human expert based visual inspection of the whole production. However, this usual solution suffers from several acute restrictions related to human operator's intrinsic limitations (reduced sensitivity for very small defects, detection exhaustiveness alteration due to attentiveness shrinkage, operator's tiredness and weariness due to repetitive nature of fault detection and fault diagnosis tasks).

To overcome these problems we have proposed a detection approach based on Nomarski's microscopy issued imaging [1] [2]. This method provides robust detection and reliable measurement of outward defects, making plausible a fully automatic inspection of optical products. However, the above-mentioned detection process should be completed by an automatic classification system in order to discriminate the "false" defects (correctable defects) from "true" (permanent) ones. In fact, because of industrial environment, a number of correctable defects (like dusts or cleaning marks) are usually present beside the potential abiding defects. That is why the association of a faults' classification system to the aforementioned detection module is a foremost supply to ensure a reliable diagnosis. In a precedent paper [3], we proposed a method to extract relevant data from raw Nomarski images. In the aim of effectively classify these descriptors, neural network based techniques seem appropriate because they have shown many attractive features in complex pattern recognition and classification tasks [4] [5]. But we are dealing with high dimensional data (13 and more components vectors) so behaviors of a number of these algorithms could be affected. To avoid this problem we are investigating different dimension reduction techniques for achieving better classification (in terms of performance and processing time).

This paper is organized as follows: in the next section, motivations for data dimensionality reduction and also Self Organizing Maps (SOM), Component Analysis (CCA) and Curvilinear Distance Analysis (CDA), three techniques to perform this task are introduced. These techniques have been tested using an experimental protocol presented in Section 3. The Section 4 deals with experiments results: first a comparison of data projections quality and an analysis of their possible impact on classification tasks are carried out. Secondly this impact is studied on a real classification problem involving Multilayer Percepton artificial neural network, and the obtained results are discussed. The Section 5 concludes this work and gives a number of perspectives.

2. DATA DIMENSIONALITY REDUCTION TECHNIQUES

It can be found in literature, lot of examples using various dimension reduction techniques (linear or not) as a preliminary step before more refined processing:, Principal Component Analysis (PCA) [6], Self Organizing Maps (SOM) [7;8], Curvilinear (CCA) [9;10] or (CDA) [11].

Dealing with high-dimensional data indeed poses problems, known as “curse of dimensionality” [10]. First sample number required to reach a predefined level of precision in approximation tasks increases exponentially with dimension. Thus, intuitively, the sample number needed to properly learn high-dimensional data becomes quickly much too large to be collected by real systems, when dimension of data increases. Moreover surprising phenomena appear when working in high dimension [12] : for example, distances variance between vectors remains fixed while its average increases with the space dimension, and Gaussian kernel local properties are also lost. These last points explain that behaviour of a number of artificial neural network algorithms could be affected while dealing with high-dimensional data. Fortunately, most real-world problem data are located in a manifold of dimension p much smaller than its raw dimension. Reducing data dimensionality to a smaller value can therefore decrease the problems related to high dimension.

2.1 SELF ORGANIZING MAPS

Self-Organizing Map is a classical method originally proposed by Kohonen [13]. The algorithm projects multidimensional feature space into a low-dimensional presentation. Typically a SOM consists of a two dimensional grid of neurons. A vector of features is associated with each neuron. During the training phase, these vectors are tuned to represent the training data under constraint of neighbourhood conservation Similar data are projected to the same or nearby neurons in the SOM, while different ones are mapped to neurons located further from each other, resulting in a clustering data. Thus SOM is an efficient tool for quantizing the data’s space and projecting this space onto a low-dimensional space, while conserving its topology. SOM is often used in industrial engineering [14], [15] to characterize high-dimensional data or to carry out classification tasks. Unfortunately it suffers of major drawbacks: first the configuration of the topology is static and should be fixed a priori (what is efficient only for little values of projection subspace dimension), moreover the method defines only a discrete nonlinear subspace and finally algorithm is computationally too expensive to be practically applied for projection space dimension higher than three.

2.2 CURVILINEAR COMPONENTS ANALYSIS

The goal of this technique proposed by Demartines [16] is to reproduce the topology of a n-dimension original space in a new p-dimension space (where p<n) without fixing any configuration of the topology. To do so, a criterion characterizing the differences between original and projected space topologies is processed:

$$E_{CCA} = \frac{1}{2}\sum_i \sum_{j \neq i} (d_{ij}^n - d_{ij}^p)^2 F(d_{ij}^p) \qquad (1)$$

where d_{ij}^n (respectively d_{ij}^p) is the Euclidean distance between vectors x_i and x_j of considered distribution in original space (resp. in projected space), and F is a decreasing function which favors local topology with respect to the global topology. This energy function is minimized by stochastic gradient descent [17]:

$$\forall i \neq j, \Delta x_i^p = \alpha(t)\frac{d_{ij}^n - d_{ij}^p}{d_{ij}^p}u(\lambda(t) - d_{ij}^p)(x_i^p - x_j^p) \qquad (2)$$

where $\alpha : \Re^+ \to [0;1]$ is a decreasing function representing a learning parameter, and $\lambda : \Re^+ \to \Re^+$ is a decreasing function too, representing a neighborhood factor. CCA provides also a similar method to project, in continuous way, new points in the original space onto the projected space, using the knowledge of already projected vectors.

2.3 CURVILINEAR DISTANCE ANALYSIS

Since CCA encounters difficulties with unfolding of very non-linear manifolds, an evolution called CDA has been proposed [18]. It involves curvilinear distances (in order to better approximate geodesic distances on the considered manifold) instead of Euclidean ones. Curvilinear distances are processed in two steps way. First is built a graph between vectors by considering k-NN, ε, or other neighborhood, weighted by Euclidean distance between adjacent nodes. Then the curvilinear distance between two vectors is computed as the minimal distance between these vectors in the graph using Dijkstra’s algorithm. Finally the original CCA algorithm is applied using processed curvilinear distances. This algorithm allows dealing with very non-linear manifolds and is much more robust against the choices of α and λ functions.

3. EXPERIMENTAL VALIDATION PROTOCOL

In order to obtain exploitable data for a classification scheme, we first needed to extract relevant information of raw Nomarski’s microscopy issued images. We proposed to proceed in two steps [2]: first a detected items’ images extraction phase and then an appropriated coding of the extracted images. The image associated to a given detected item is constructed considering a stripe of ten pixels around its pixels. Thus the obtained image gives an isolated (from other items) representation of the defect (e.g. depicts the defect in its immediate environment). Fig. 1 gives four examples of detected items’ images using the aforementioned technique. It shows different characteristic items which could be found on optical device in industrial environment. The information contained in such images is highly redundant. Furthermore, the generated images don’t have necessarily the same dimension (typically this dimension can turn out to be thousand times as high). That is why these raw data (images) cannot be directly processed and has to be appropriately encoded.

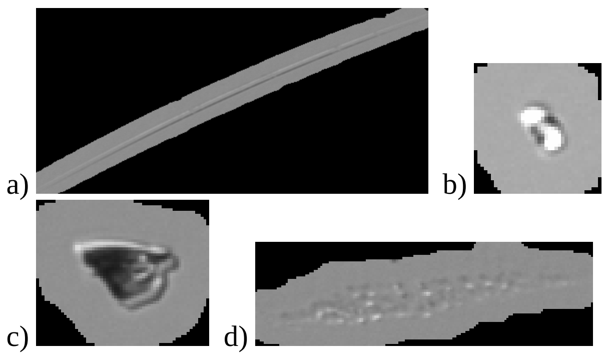


Fig. 1 – Images of characteristic items: a) scratch; b) dig; c) dust; d) cleaning marks.

This is done using a set of Fourier-Mellin transform issued invariants described bellow. The Fourier-Mellin transform of a function $f(r;\theta)$, in polar coordinates, is given by relation (1), with $q \in Z$, $s = \sigma + ip \in C$ (see[19]):

$$M_f(q;s) = \int_{r=0}^{\infty}\int_{\theta=0}^{2\pi} r^{s-1}\exp(-iq\theta)f(r;\theta)drd\theta \qquad (3)$$

In [20], are proposed a set of features invariant on geometric transformations:

$$I_f(q;s) = M_f(q;s)\left[M_f(0;\sigma)\right]^{-\frac{s}{\sigma}}\left[M_f(1;\sigma)\right]^{-q}\left|M_f(1;\sigma)\right|^{q} \qquad (4)$$

In order to validate the above-presented concepts and to provide an industrial prototype, an automatic control system has been realized. It involves an Olympus B52 microscope combined with a Corvus stage (see Fig. 2), which allows scanning an entire optical component. 50x magnification is used, that leads to microscopic 1.77mm x 1.33 mm fields and 1.28µm x 1.28µm sized pixels.

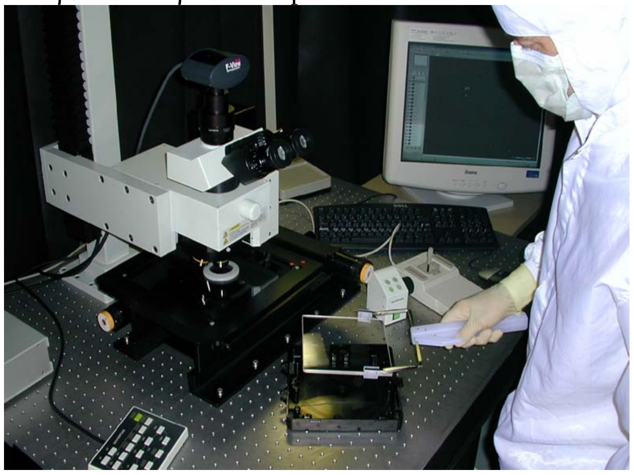


Fig. 2 – The Olympus microscope.

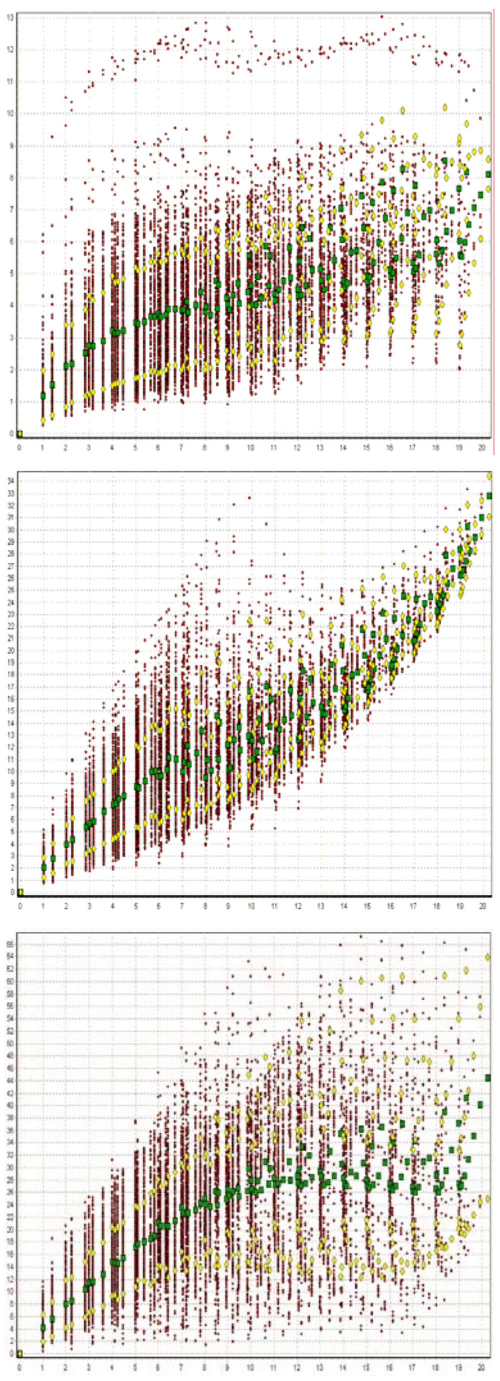


Fig. 3 – dy-dx representation of the three obtained SOMs for database B (mean □ and standard deviation ◊ of dx are also represented). Top: SOM; middle: CCA; bottom: CDA.

The defects detection methods are implemented in the system in an efficient way, so the prototype permits to check optical devices within execution times in adequacy with production conditions (typically 10 mm x 10 mm surface measured in one minute). These facilities were used to acquire a great number of defects images. These images were coded using Fourier-Mellin transform with $\sigma = 1$ and $(q,p) \in \{(q,p)/(q=0;0 \le p \le P) \cup (1 \le q \le Q;-P \le p \le P)\}$ where $P=1$ and $Q=2$ (see Equation 3 and 4). Such orders of Mellin resp. Fourier spectrum (P resp. Q values) form a first compromise between the size and the quality of the representation and provides a set of 13 features for each item. Three experiments called A, B, C were carried out, using two optical devices. Table 1 shows the different parameters corresponding to these experiments. It's important to note that, in order to avoid false classes learning, items images depicting microscopic field boundaries or two (or more) different defects are discarded from used database. First, since database C is issued from a cleaned device, it's constituted with almost only "permanent" defect. And because database B came from the measurement of the same optical device but without cleaning phase, it's constituted with the same type of "permanent" defects but also with "abiding" ones.

Table 1. Description of the three experiments supplying studied databases.

Database	A	B	C
Optical Device Identifiant	1	2	2
Cleaning	NO	NO	YES
Number of studied microscopic fields	1178	605	529
Correspondant studied area	28	14	12.5
Number of items in the learning database	3865	1910	1544

In the aim of studying structure of space described by database when reducing its dimension, we perform some experiments. First a reduction of dimensionality from 13 (raw dimensionality) to 2 of the database B was performed using SOM, CCA and CDA, in order to compare projection quality of these three techniques. Then the entire database C was projected into the obtained space in order to evaluate the pertinence of dimensionality reduction for discrimination between "correctable" and "abiding" defects. Secondly, a synthetic classification task, involving aforementioned databases and Multilayer Perceptron artificial neural network, was carried out with and without dimensionality reduction phase with the aim to demonstrate usefulness of such pre-processing phase. Finally, we validate the previous results by studying the impact of different dimensionality reduction on a real problem: an expert was asked to define two different real classes of defects and a MLP was used to discriminate between these two classes.

4. EXPERIMENTAL RESULTS AND ANALYSIS

4.1 QUALITY OF PROJECTION

Dimensionality reduction has been performed using the three aforementioned techniques, SOM, ACC and CDA on database B. To compare the results of the three experiments, the 2-D projections issued from CCA and CDA were processed by a SOM, using the same shape of grid (20x8) as in the SOM experiment. An important point is that SOM is just used, in these two past cases, to perform a quantization and not for dimension reduction, since it works on a 2 dimension space. Therefore, we can directly compare dimension reduction ability of the different techniques by comparing these maps with map obtained by applying SOM's algorithm on raw data. The quality evaluation of non-linear projection of the data space onto the neurons grid space is performed by studying, for each pair of neurons, the dx distance between these two neurons in the data space, versus the dy distance between these two neurons in the grid space [21]. For each couple of neurons $(i;j)$ we draw a point $(dy(i,j);dx(i,j))$ where $dx(i,j)=\left\|\vec{x}_i-\vec{x}_j\right\|$ and $dy(i,j)=\left\|\vec{y}_i-\vec{y}_j\right\|$. $\vec{x}_k$ (resp. $\vec{y}_k$) is the vector of features corresponding to the k-th neuron in the data space (resp. in the grid space). If the topology of the data space is not well respected, dx is not related to dy and we obtain a diffuse cloud of points. On the contrary, if neurons organization is correct, the drawn points are almost arranged along a straight line.

First, in Fig. 3, cloud of points is more diffuse for SOM than in the case of CCA, and the curve constituted by dx averages for each dy less uniformly monotonic. It reveals the fact that the CCA performs better than SOM, while approximately the same quantity is minimized. The cloud obtained for CDA is quite different because dy is related to curvilinear distance and not Euclidean one. The figure is however the same as for CCA for little dy value, because in these cases Euclidean distance is a good approximate of curvilinear one (and therefore distribution is locally linear).

4.2 ANALYSIS OF POSSIBLE IMPACT ON CLASSIFICATION TASKS

We now consider the database C (only "permanent" defects) and project its items onto the three previously obtained SOMs. We perform also an equivalent experiment on raw data (13-dimension), using k-means algorithm with k=20x8=160. Since k-means algorithm has identical behaviour as SOM, except concerning neighbourhood constraints, it has the same effect on projected items distribution but doesn't allow visual representation. Projected items distribution after SOM (Fig. 3), CCA (Fig. 4) and CDA (Fig. 5) dimension reduction are studied. In these figures, the equalized grey level depicts the number of projected items for each SOM's cell (this number is also reported in the cell).

In table 2 are reported some characteristic values of distributions "homogeneity": entropy and standard deviation of projected items number in each cell; number of empty or quasi-empty cells (less than three projected items). Maps and numerical measurements for SOM and CCA are comparable and therefore these techniques are equivalent for the considered problem. CCA is however easier to perform (no a priori knowledge or difficult choice) and provide more information (continuous projection). CDA offers the same advantages as CCA, but it seems to be more appropriate for pre-processing before classification. Corresponding map depicts indeed more specific "areas" for database C projected defects.

Table 2. Different measurements characterizing the projections distribution of database C items (permanent defects).

Dimensionality reduction technique	None	SOM	CCA	CDA
Standard-deviation of distribution	8.72	5.78	5.72	7.04
Entropy of defects distribution	2.055	2.114	2.121	2.088
Number of empty cells	15	9	5	7
Number of cells with less than 3 defects	30	26	20	32

This intuition is confirmed by numerical measurements: entropy is lower than in SOM and CCA cases (better organization), standard deviation is higher (better contrast between full and empty areas) and there are more quasi-empty cells. We think that this organization is a foremost guarantee for the dimension reduction to allow a better classification. We can also remark that results obtained with CDA are fairly similar as those with raw data; it shows that little information is lost while reducing dimensionality.

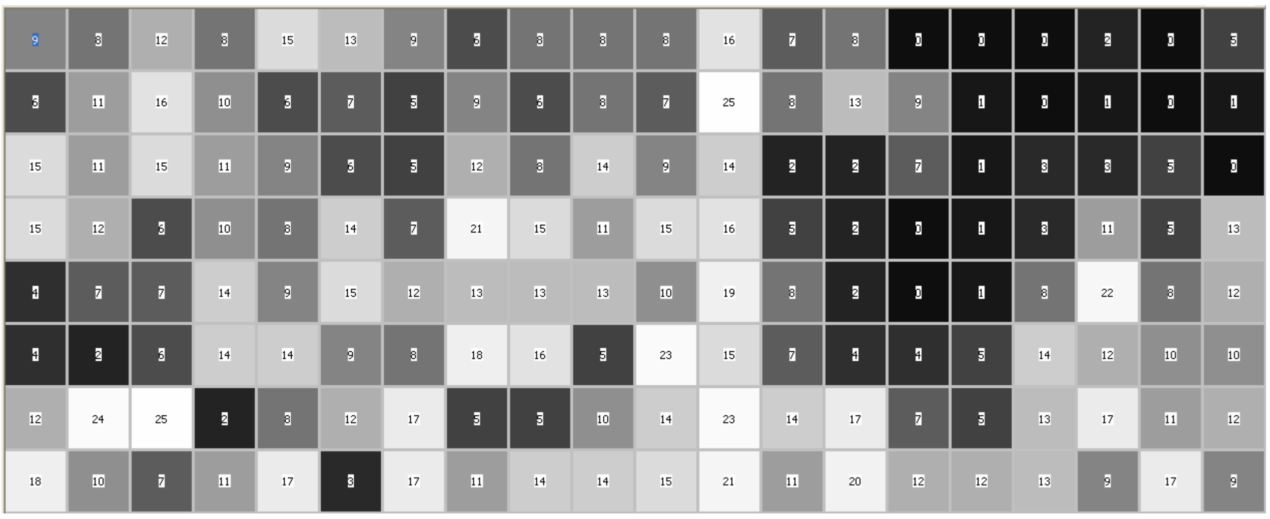


Fig. 4. Distribution of projected items in SOM map. (SOM reduction dimension)

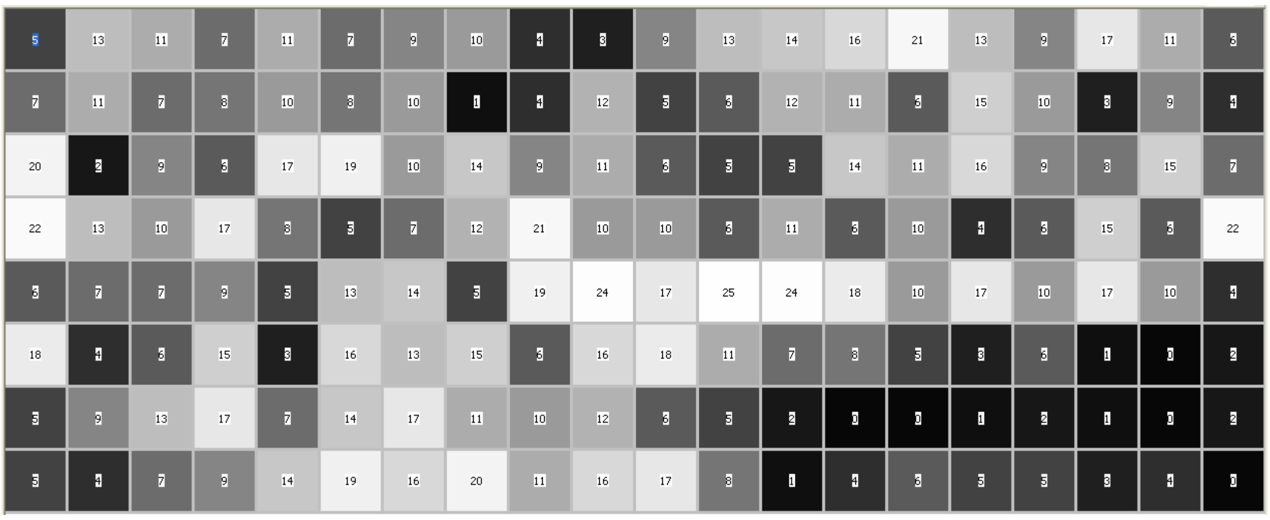


Fig. 5. Distribution of projected items in SOM map. (CCA reduction dimension)

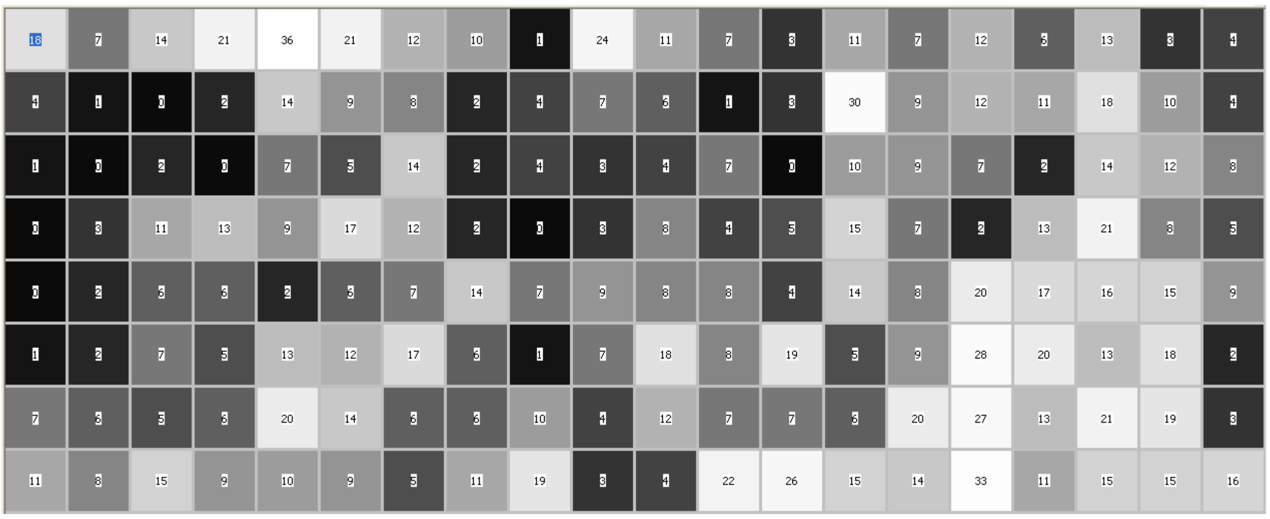


Fig. 6. Distribution of projected items in SOM map. (CDA reduction dimension)

4.3 EXPERIMENTATION ON A REAL PROBLEM

In order to evaluate the benefits of using dimensionality reduction in our diagnosis system, we performed a series of experiments involving a real classification task. Items of the databases A and B were labelled by an expert with two different labels: “dust” (class 1) and “other defects” (class 2). Considering our final goal, this is the most important distinction which the diagnosis system must be able to do. Table 3 shows items repartition between the two defined classes.

Table 3. Description of real classification databases.

Database	Coming from database	Total number of items	Label 1 items	Label 2 items
1	A	3865	275	3590
2	B	1910	184	1726

Using these databases, a number of experiments were carried out, in accordance with a same procedure. It involved a multilayer perceptron with n input neurons, 35 neurons in one hidden layer, and 2 output neurons (n-35-2 MLP). First this artificial neural network was trained for discrimination task between classes 1 and 2, using database B. This training phase used BFGS (Broyden, Fletcher, Goldfarb, and Shanno) with Bayesian regularization algorithm, and was achieved 5 times. Subsequently, the generalization ability of obtained neural network was processed using database A. Since database A and B issued from different optical devices, such generalization results are significant. Following this procedure, 28 different experiments were conducted with the aim of studying the global classification performance and the impact of SOM, ACC and CDA dimensionality reduction on this performance. First experiment used original Fourrier-Mellin issued features (13-dimensional), the second used 2D SOM reduced features, and the others used the original features after ACC or CDA n-dimensional space reduction (with n varying between 2 and 13).

Fig. 7 depicts classification global performances (calculated by averaging percentage of well-classified items for the 5 trainings) for the 28 different experiments (for different CDA and CCA issued data dimensionality reduction, for 2D SOM data dimensionality reduction and using raw data). Fig. 8 and Fig. 9 show the class 1 (“dust defects”) and the class 2 (“other defects”) classification performances respectively. In this two figures the standard deviation of the 5 experiments results are also represented.

4.4 DISCUSSION

First, these experiments show that CDA and CCA generate almost the same performances (CDA is although slightly better) but outperform SOM issued results (see Table 4) for 2D data dimensionality reduction. It is due to the better quality and the continuous nature of the projection provided by these techniques. Thus, it confirms the previous results presented in this paper.

Table 4. Classification performances for the three different techniques, when reducing data dimensionality down to 2.

Dimensionality Reduction Technique	SOM	CCA	CDA
Class 1 Recognition	63.9 %	61.9 %	56.8 %
Class 2 Recognition	91.6 %	96.9 %	97.7 %
Global Performance	**89.7 %**	**94.4 %**	**94.8 %**

Secondly, we can remark (see Table 5) that equivalent performances can be obtained using low-dimensional data instead of unprocessed defects representations (for example using 6-dimensional CDA or 8-dimensional CCA issued representation instead of raw 13-dimensional representation).

Table 5. Classification performances with raw data, CDA and CCA issued data. For CCA and CDA, results are given for the data dimensionality which allows the better global performance.

Used Data	Raw	CCA Issued	CDA Issued
Data Dimensionality	13	8	6
Class 1 Recognition	70.2 %	63.9 %	62.0 %
Class 2 Recognition	97.2 %	97.2 %	97.7 %
Global Performance	**95.3 %**	**94.8 %**	**95.1 %**

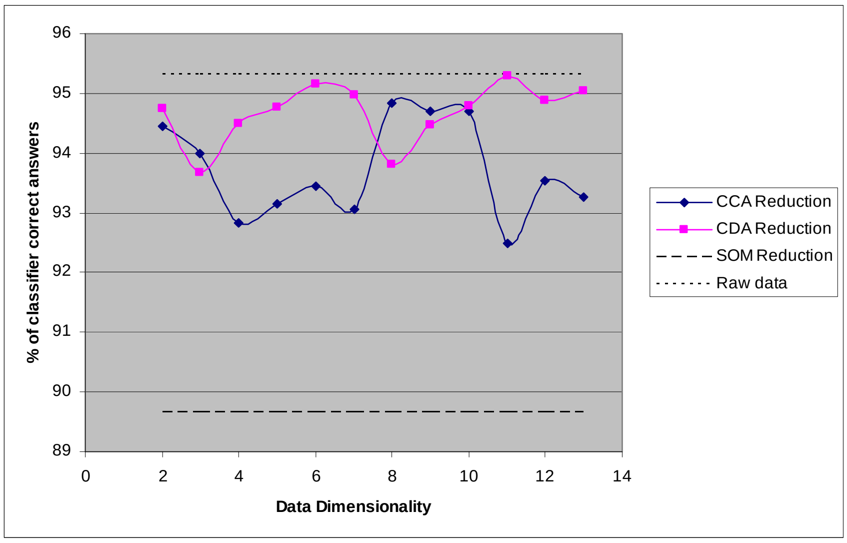


Fig. 7. Classification global performances.

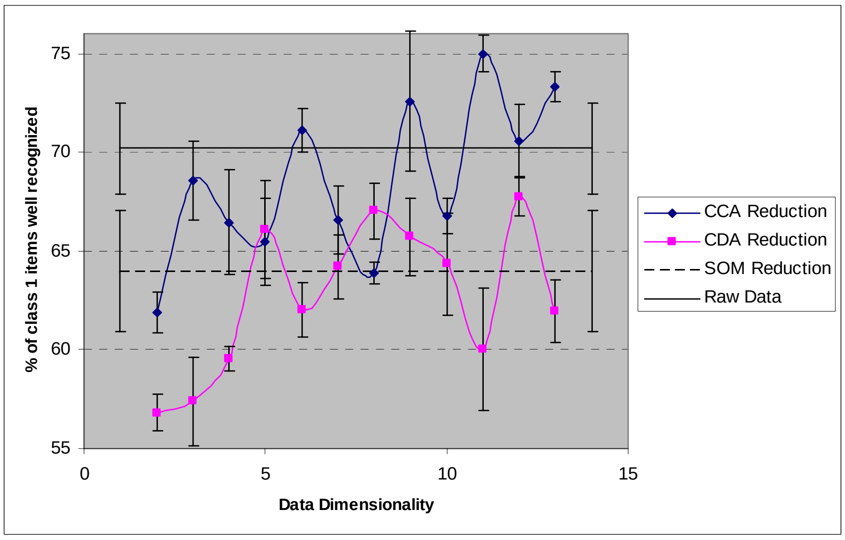


Fig. 8. Class 1 (“dust defects”) classification performances. Standard deviation of the 5 experiments results are also represented.

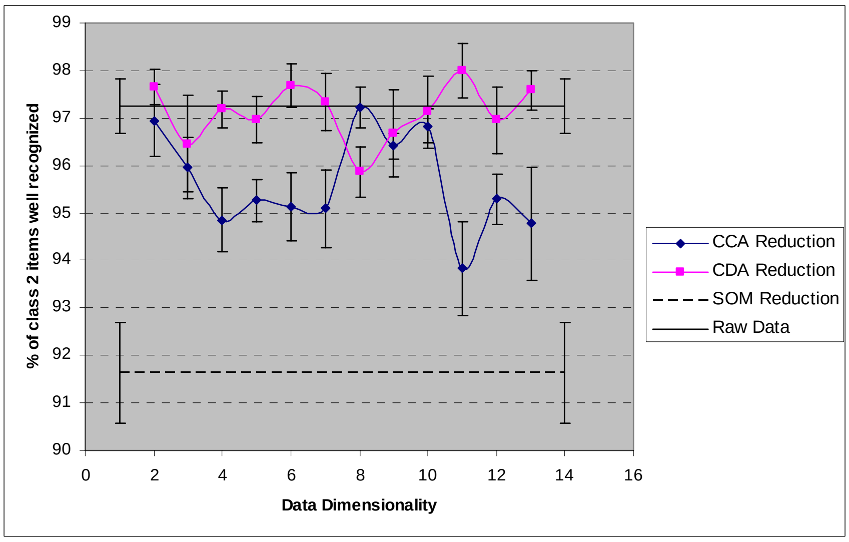


Fig. 9. Class 2 (“other defects”) classification performances. Standard deviation of the 5 experiments results are also represented.

As a consequence neural architecture complexity and therefore processing time can be saved using dimensionality reduction, while keeping performance level. Moreover, obtained scores are satisfactory: about 65% of "dust" defects are well-recognized (this can be enough for aimed application) as well as about 97% of other defects (the few 3% errors can however pose problems because every "permanent" defect has to be reported). Furthermore, we think that this significant performances difference between class 1 and class 2 recognition performances is due to the fact that class 1 is underrepresented in learning database. On another hand, obtained results show that the compromise between these two different performances can be managed by choosing an adequate data final dimension.

Finally, we can see in Fig. 8 and 9 that standard deviation values are relatively small, what's a sign of the training phase quality. In addition, performance in class 1 recognition is relatively low when reducing data dimensionality down to two (there's probably no more enough information to correctly classify "dust" defects). But it increases steadily until 5-dimensional CDA issued space. This result agrees with estimated items distribution intrinsic dimension between 5 and 6 (estimation used Grassberger-Procaccia [22] modified algorithm [23]). It's interesting to note that the global performances are optimal for such a final dimensionality using CDA (about 95% of well-classified defects what is equivalent to performances obtained when using raw data). This level of performance is reached as well with 10 and 11-dimensional CDA reduction and only in these cases. This last point will be studied in greater detail in another paper.

5. CONCLUSION

A reliable diagnosis of aesthetic flaws in high-quality optical devices is a crucial task to ensure products' nominal specification and to enhance the production quality by studying the impact of the process on such defects. To ensure a reliable diagnosis, an automatic classification system is needed in order to discriminate the "false" defects (correctable defects) from "abiding" (permanent) ones. Unfortunately relevant data extracted from raw Nomarski image during defects detection phase are high dimensional. This can have harmful effect on behaviors of artificial neural networks which are suitable to perform such a challenging classification. Reducing the dimension of the data to a smaller value can decrease the problems related to high dimension. In this paper we have compared different techniques, SOM, CCA and CDA which permit such dimensionality reduction and evaluated their possible impact on classification tasks involving real industrial data. CDA seems to be the most suitable technique and we have demonstrated its ability to enhance performances (in terms of time and/or well-classified items) in a real classification problem. Next phase of this work will deal with classification tasks involving more classes. We want also use much more Fourier-Mellin invariants, because we think that it would improve classification performance by supplying additional information. CDA based dimensionality reduction technique would in this case be a foremost step to keep reasonable classification system's complexity.

6. REFERENCES

[1] M. Voiry, F. Houbre, V. Amarger, and K. Madani. Toward Surface Imperfections Diagnosis Using Optical Microscopy Imaging in Industrial Environment. *Proceedings of the Workshop IAR & ACD 2005*, Mulhouse, France 16-18 November 2005, pp. 139-144.

[2] M. Voiry, V. Amarger, K. Madani, and F. Houbre. Combining Image Processing and Self Organizing Artificial Neural Network Based Approaches for Industrial Process Faults Clustering. *Proceedings of 13th International Multi-Conference on Advanced Computer Systems (ACS 2006)*, Miedzyzdroje, Poland 18-20 October 2006, pp. 129-138.

[3] M. Voiry, K. Madani, V. Amarger, and F. Houbre. Toward Automatic Defects Clustering in Industrial Production Process Combining Optical Detection and Unsupervised Artificial Neural Network Techniques. *Procedings of the 2nd International Workshop on Artificial Neural Networks and Intelligent Information Processing (ANNIIP 2006)* ,Setùbal, Portugal August 2006, pp. 25-34.

[4] G. P. Zhang. Neural Networks for Classification: A Survey. *IEEE Trans. on Systems, Man, and Cybernetics - Part C: Applications and Reviews* 30 (4) (2000). p. 451-462.

[5] M. Egmont-Petersen, D. de Ridder, and H. Handels. Image Processing with Neural Networks - A Review. *Pattern Recognition* 35 (2002). p. 2279-2301.

[6] P. J. Grother. Karhunen Loève Feature Extraction for Neural Handwritten Character Recognition. *Proceedings of SPIE, vol. 1709, no. Applications of Artificial Neural Networks III*, pp. 155-166, 1992.

[7] K. Boehm, W. Broll, and M. Sokolewicz. Dynamic Gesture Recognition Using Neural Networks; A Fundament for Advanced

Interaction Construction. *Proceedings of SPIE, vol. 2177, no. Stereoscopic Displays and Virtual Reality Systems*, pp. 336-346, 1994.

[8] J. Lampinen and E. Oja. Distortion Tolerant Pattern Recognition Based on Self-Organizing Feature Extraction. *IEEE Trans. On Neural Networks* 6 (3) (1995). p. 539-547.

[9] S. Buchala, N. Davey, T. M. Gale, and R. J. Frank. Analysis of Linear and Nonlinear Dimensionality Reduction Methods for Gender Classifcation of Face Images. *International Journal of Systems Science* 14 (36) (2005). p.931-942.

[10] M. Verleysen. Learning high-dimensional data. *NATO Advanced Research Workshop on Limitations and Future Trends in Neural Computing (LFTNC'2001)*, Siena, Italy 22-24 October 2001, pp.22-24.

[11] M. Lennon, G. Mercier, M. C. Mouchot, and L. Hubert-Moy. Curvilinear Component Analysis for Nonlinear Dimensionality Reduction of Hyperspectral Images. *Proceedings of SPIE, vol. 4541,Image and Signal Processing for Remote Sensing VII*, pp. 157-168, 2001.

[12] P. Demartines. *Analyse de Données par Réseaux de Neurones Auto-Organisés.* PhD Thesis Institut National Polytechnique de Grenoble, 1994.

[13] T. Kohonen. *Self Organizing Maps*. 3rd edition ed. Berlin: Springer, 2001.

[14] T. Kohonen, E. Oja, O. Simula, A. Visa, and J. Kangas. Engineering Applications of the Self-Organizing Maps. *Proceedings of the IEEE*, vol. 84, no. 10, pp. 1358-1384, Oct.1996.

[15] J. Heikkonen and J. Lampinen. Building Industrial Applications with Neural Networks. *Proceedings of European Symposium on Intelligent Techniques (ESIT'99)*, Chania, Greece June 1999.

[16] P. Demartines and J. Hérault. Vector Quantization and Projection Neural Network. *Lecture Notes in Computer Science* 686, International Workshop on Artificial Neural Networks (IWANN'93), p. 328-333.

[17] P. Demartines and J. Hérault.CCA : "Curvilinear Component Analysis". *Proceedings of 15th workshop GRETSI (GRETSI'95)*, Juan-les-pins, France, 15 September 1995.

[18] J. A. Lee, A. Lendasse, N. Donckers, and M. Verleysen. A Robust Nonlinear Projection Method. *Proceedings of European Symposium on Artificial Neural Networks (ESANN'2000)*.

[19] S. Derrode. *Représentation de Formes Planes à Niveaux de Gris par Différentes Approximations de Fourier-Mellin Analytique en vue d'Indexation de Bases d'Images.* Phd Thesis Université de Rennes I, 1999.

[20] F. Ghorbel. A Complete Invariant Description for Gray Level Images by the Harmonic Analysis Approach. *Pattern Recognition* 15 (1994). p. 1043-1051.

[21] P. Demartines and F. Blayo. Kohonen Self-Organizing Maps: Is the Normalization Necessary? *Complex Systems* 6 (2) (1992). p. 105-123.

[22] P. Grassberger and I. Procaccia. Measuring the strangeness of strange attractors. *Physica* 9 (1983). p. 189-208.

[23] F. Camastra and A. Vinciarelli. Intrinsic Dimension Estimation of Data: An Approach Based on Grassberger-Procaccia's Algorithm. *Neural Processing Letters* 14 (1) (2001). p. 27-34.

Matthieu Voiry *graduated in 2002 from ENSEIRB and received his MS degree in 2003 from Bordeaux 2 University.*

Currently he is working toward his Ph.D. at Image, Signal and Intelligent Systems Laboratory (LISSI / EA 3956) of PARIS XII University with Prof. Kurosh Madani, in collaboration with SAGEM REOSC Company.

His main research interests are image processing and artificial neural networks in industrial computer aided diagnosis.

Prof. Kurosh Madani *Received his Ph.D. degree in Electrical Engineering and Computer Sciences from University PARIS XI, Orsay, France, in 1990. From 1989 to 1990, he worked as assistant professor at Institute of Fundamental Electronics of PARIS XI University. In 1990, he joined Creteil-Senart Institute of Technology of University PARIS XII – Val de Marne, Lieusaint, France, where he worked from 1990 to 1998 as assistant professor. In 1995, he received the DHDR Doctor Habilitate degree (senior research Dr. Hab. degree) from University PARIS XII – Val de Marne. Since 1998 he works as Chair Professor in Electrical Engineering of Senart Institute of Technology of University PARIS XII. From 1992 to 2004 he has been head of Intelligence in Instrumentation and Systems Laboratory (I2S / JE 2353) located at Senart Institute of Technology. Since 2005, he is head of one of the three research groups of Image, Signal and Intelligent Systems Laboratory (LISSI / EA 3956) of PARIS XII University. He has worked on both digital and analog implementation of processors arrays for image processing, electro-optical random number* *generation, and both analog and digital ANN implementation.*

His current research interests include large ANN structures modeling and implementation, hybrid neural based information processing systems and their software and hardware implementations, design and implementation of real-time neuro-control and neural based fault detection and diagnosis systems. Since 1996 he is a permanent member (elected Academician) of International Informatization Academy. In 1997, he was also elected as Academician of International Academy of Technological Cybernetics.

***Dr. Véronique Amarger** received her PhD degree in Microelectronics and Computer Science from the University of Paris 7, France, in 1993. Since 1993, she has joined Senart Institute of Technology of PARIS XII – Val de Marne University, one of two Institutes of Technology of this University, where she works as Assistant Professor. She is a Staff Member of Image, Signal and Intelligent Systems Laboratory (LISSI / EA 3956) of PARIS XII University.*

Her main research interests concern the field of Bio-inspired Artificial Intelligence, Neural Networks and Computer Aided Diagnosis Systems design and applications.

Joël Bernier graduated from the French Engineering school Ecole Superieure d'Optique (ESO) in 1979, he worked 15 years at MATRA, now part of EADS group, assigned to project management for Optronic equipment development. Then, he join the SFIM group and in particular its subsidiary REOSC in 1997 where he was the Astronomy and Laser business unit manager. After the takeover of SFIM by SAGEM SA in 1999, he is in charge of the R&D team of the REOSC Program, where all high performance optics for astronomy, laser, space or lithography are designed and manufactured.

computing@computingonline.net
www.computingonline.net

ISSN 1727-6209
International Scientific
Journal of Computing

ESTIMATING COMPLEXITY OF CLASSIFICATION TASKS USING NEUROCOMPUTERS TECHNOLOGY

Ivan Budnyk, Abdennasser Chebira, Kurosh Madani

Images, Signals and Intelligent Systems Laboratory (LISSI / EA 3956)
PARIS 12 – Val de Marne University, Senart-Fontainebleau Institute of Technology,
Bat. A, Av. Pierre Point, F-77127 Lieusaint, France,
{ivan.budnyk, chebira, madani}@univ-paris12.fr,
http://lissi.univ-paris12.fr

Abstract: *This paper presents an alternative approach for estimating task complexity. Construction of a self-organizing neural tree structure, following the paradigm "divide and rule", requires knowledge about task complexity. Our aim is to determine complexity indicator function and to hallmark its' main properties. A new approach uses IBM © Zero Instruction Set Computer (ZISC-036 ®) and applies for a range of the different classification tasks.*

Keywords: *IBM © Zero Instruction Set Computer (ZISC-036 ®) Neurocomputer, Neural tree modular architecture, T-DTS, DNA (Deoxyribonucleic acid), RNA (Ribonucleic acid), exon, intron, Splice junctions problem, Tic-tac-toe endgame problem.*

1. INTRODUCTION

In this paper we present an alternative complexity estimating approach for a modular neural tree structure. This structure uses key module of the complexity estimation for solving classification problem following the paradigm "divide and rule".

This general modular tree structure [1] is Tree Divide To Simplify (T-DTS) Fig. 1. Complexity reduction is the key point on which the modular approach acts. Complexity reduction performs not only at the problem's solution level but also at the processing procedure's level. The main idea is to reduce the complexity by splitting a complex problem into a set of simpler sub-problems: this leads to "multi-modeling" where a set of simple models is used to sculpt a complex behavior. Thus, one of the foremost functions to be performed is the complexity estimation.

We introduce in this paper the complexity which is based on ZISC-036 ® neurocomputer [2]. Before describing the proposed approach, we present in the second section T-DTS paradigm and the hardware tool used for complexity estimation, IBM © Zero Instruction Set Computer (ZISC-036 ®).

Third section properly contains the description of *a new approach*. *A validation* and *a definition of the classification complexity* contain section four. Obtained results and their overview are presented in the sections five. Final section presents conclusion and further perspectives of the work.

2. THE NEURAL STRUCTURE AND NEUROCOMPUTER

In a very large number of cases dealing with real world dilemmas and applications (system identification, industrial processes, manufacturing regulation, optimization, decision, pattern recognition systems, plants safety, etc), information is available as data stored in databases [3]. An efficient data processing becomes a chief condition to solve problems related to above-mentioned areas.

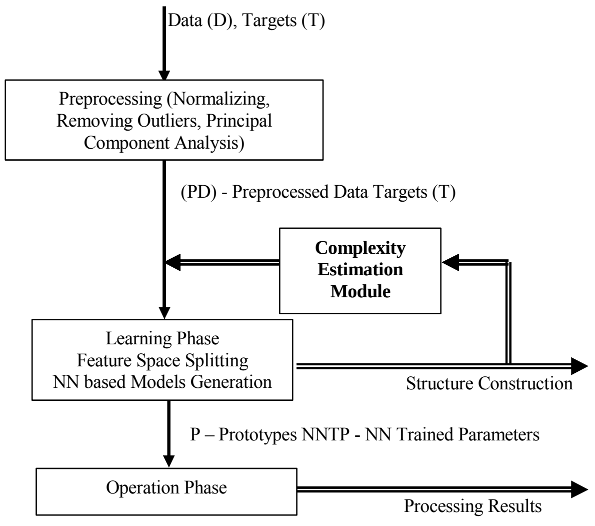
The use of machine learning approaches for such problems can be justified in the following way [4]:

- machine learning approaches produce predictable models with comparable and often superior quality than models based on the statistical analysis.
- they are more easy to understand, intelligible to human beings.
- instead of trying to fit the data to the model, most of machine learning approaches build models by including a knowledge that will accommodate all cases in the sample population. Real life models can be sometimes approximated with mathematical models (liner or non-linear), but sometimes that is not possible.
- machine learning approaches offer better solution when the knowledge describing the real life

world is incomplete, inexact, and imprecise.

An issue of using machine learning approaches is a capability to model complexity reduction by splitting a complex problem into a set of simpler sub-problems: multi-modeling, where a set of simple models, is used to sculpt a complex behavior [5] [6]. For such purpose, a tree-like splitting process, based on complexity estimation, divides the problem's representative database on a set of sub-databases, constructing a specific model (dedicated processing module) for each sub-database. That leads to a modular tree-like processing architecture including several models.

In order to deal with real word problem, we propose, a modular approach based on *divide and conquer* paradigm [1] [3]. In this approach, Tree Divide To Simplify or T-DTS, we divide a problem into sub-problems recursively and generate a neural tree computing structure.



Data (D), Targets (T)
Preprocessing (Normalizing, Removing Outliers, Principal Component Analysis)
(PD) - Preprocessed Data Targets (T)
Complexity Estimation Module
Learning Phase Feature Space Splitting NN based Models Generation
Structure Construction
P – Prototypes NNTP - NN Trained Parameters
Operation Phase
Processing Results

Fig. 1 - General bloc diagram of T-DTS.

T-DTS and associated algorithm(s) construct(s) a tree-like evolutionary neural architecture automatically where nodes, call "Splitting Units", are decision units and leafs, call "Neural Network based Models", correspond to neural based processing units [6] [7] [8].

T-DTS includes two main operation modes. The first is the learning phase, when T-DTS system decomposes the input database and provides processing sub-structures, and tools for decomposed sets of data. The second phase is the operation phase. Fig. 1, gives the general bloc diagram of T-DTS operational steps.

Fig. 1 shows that T-DTS could be characterized by four main blocks: "data pre-processing", "learning process", "generalization process", "complexity estimation module". The tree structure construction is based mainly on the complexity estimation module. This module introduces a feedback in the learning process and control the tree building process. The reliability of tree model to sculpt the problem behavior is associated mainly to the complexity estimation module. This work focuses on the aspect of complexity estimation and proposes *a new approach* based on neurocomputer hardware ZISC-036 ®:

IBM © ZISC-036 ® neuron-computer is a fully integrated circuit based on neural network designed for recognition and classification [2] [6] [9]. It is a parallel neural processor based on the Reduced Coulomb Energy (RCE) [10] and K-Nearest Neighbor (KNN) [11] algorithms. Each chip is the implementation of the RBF-like model [12].

RBF approach could be seen as mapping an N-dimensional space by prototypes. Each prototype is associated with a category and an influence field. ZISC-036 ® system implements two kinds of distance metrics that we have used L1 and LSUP respectively. The first one, L1 corresponds to a polyhedral volume influence field and the second LSUP - to a hyper-cubical one.

During estimation complexity the RCE is used. This hardware implemented method on ZISC-036 ® is effective in separating patterns classes by nonlinear boundaries. However, the RCE network depends on the user-specified parameters which are computationally expensive to optimize [10]. Each ZISC-036 ® neuron of the network is an element, which is able to:

memorize a prototype composed of 64 components, the associated category, an influence field and a context,

compute the distance and compare basing on the selected norm L1/LSUP between its memorized prototype and the input vector,

interact with other neurons adjusting their influence fields during learning phase in order to find the minimal distance, category, etc.

The next section presents *a complexity estimation approach* that is based on such neurocomputer's capabilities.

3. COMPLEXITY ESTIMATION APPROACH

It is clear that the efficiency of neural network models is the basic precondition of their practical applicability in the artificial intelligence. With respect to the fact that neural networks models were inspired by living organisms which perform the relevant function efficiently, this approach leads to the complexity-theoretic definition of intelligence: the way of efficient knowledge representation.

We can understand the efficiency in three senses: an efficient creation and adaptation of this representation (learning complexity), its memory demands (descriptive complexity) and efficient knowledge retrieval (computational power). [13]

The complexity estimating in T-DTS is used to understand the behavior of classifiers. A chief aim of the complexity estimating is to check, to measure the difficulty of a classification task before proper processing and to construct an optimized modular tree-like system.

The definition of the classification complexity as a complexity in general term relates the difficulty in formalization of the whole compared to that of its fundamental parts (from the point of view of the language). It is only applicable in cases where there is at least a possibility of gaining almost complete information about the components, thus clearly separating ignorance from complexity. We have different concepts of complexity depending on the base language chosen, the type of difficulty focused on and the type of formulation desired within that language [14].

In our work we determine a complexity as the amount of computational resource that it takes to solve a classification problem. Thus, *a complexity* here is the related to the amount of the resource supplied.

Thus way of defining complexity as *a computing complexity* of classification is adopted for our approach, because of hardware limitation of the classification tools [15].

Supposing a classification problem has a collection of *m* objects of database associated to labels/categories. We classify and estimate the classification complexity using the neurocomputer without regard to a classifier.

Firstly, we learn the ZISC-036 ® neurocomputer to classify objects using the associated database. Then estimate the task computational complexity, analyzing the generated ZISC-036 ® neural network structure that has been created by this neurocomputer. In general, we expect that a method which satisfying demands of the classification method *will involve a more complex structure for a more complex problem*, or being more precisely, *the neural network structure will be an archetypal platform for extracting underlying properties/parameters of the classification complexity* [16].

The simplest neural network structure feature is the number *n* of neurons created during the *learning* phase. The following indicator is defined (1), where parameter *n* is a value that reflects complexity and *m* – database size that have been used to train neural network structure:

$$Q = \frac{n}{m}\ ,\ m \geq 1, n \geq 0 \qquad (1)$$

We suppose that there exists some function of complexity *n=g(.)* ,where the arguments of it may be the signal-to-noise ratio, the dimension of the representation space, boundary non-linearity and/or database size.

In a first approach, we consider only *g(.)* function's variations according to *m* axis: *g(m)*. We suppose that our database (e.g. the used database) is free of any incorrect or missing information.

On the basis on $g_p(m)$, where *p* is vector of parameters, a complexity indicator Q_p defines:

$$Q_p(m) = \frac{g_p(m)}{m}\ , m \geq 1, g_p(m) \geq 0 \qquad (2)$$

We expect that for the same problem, as we enhance *m*, the problem seem to be less complex: more information reduces problem ambiguity. On the other hand, for problems of different and increasing complexity, an evolution of Q_p indicator should have a relevant trend.

Also we can interpret obtained structure of neurons as the result of *computational process* [17]. The process consists of a *program* plus *data.* This idea underlies efforts to define both classical algorithmic complexity (eg. Chaitin, Kolmogorov [13]) and information entropy (eg. Papentin, Brooks).

Most definitions of this kind (eg. Gramma, Bennett's and Logren's complexity [14]) hinge on the notion of *the shortest program.* This idea is unworkable in practice because in general we cannot prove that a particular program is the shortest. An alternative point of our approach is a computing of the complexity in the context [17]. Of course, this is another definition of computational complexity [18]. However, the strong feature of this definition is a strong orientation on the limitation of computational capabilities.

In order to check the behavior of the *indicator-function* (2), we have defined a specific bench-mark and applied extracted approach to DNA, Tic-tac-toe classification problems present in the following sections.

Specific benchmarking database.

Basically, we construct 5 databases representing a mapping of a restricted 2D space to 2 categories, Fig. 2. Each pattern is divided into two and more equal striped sub-zones, each of them belonging to the categories 1 or 2 alternatively.

In *learning* phase we create samples using randomly generated plots (objects) with coordinates *(x,y).* The number of samples *m*, in our case of uniform random distribution, naturally has an influence on the quality of the striped zones (categories) demarcation. According to the value of the first coordinate *x*, and according to the amount of the striped sub-zones, the appropriate category *c* is assigned to the sample, and such structure (x_j, y_j, c_j) sends to neurocomputer on the *learning*.

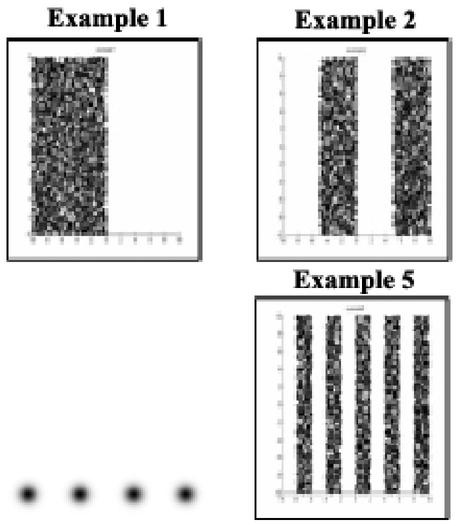


Fig. 2 – Test patterns.

The second phase is *classification* or in the other words real *testing* of the generalizing ZISC-036 ® neurocomputer abilities. We again, randomly and uniformly, generate *m* samples and their associated category. Gaining classification statistics, we compute the indicator-function Q_p.

DNA patterns classification problem.

Deoxyribonucleic acid (DNA) is a nucleic acid that contains the genetic instructions for the development and function of living organisms. This is two long strands entwine like vines, in the shape of a double helix.

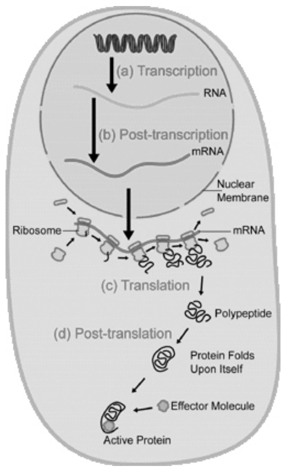


Fig. 3 – Genes DNA (black) transcription into RNA in the nucleus of the cell.

The major function of DNA is to encode the sequence of amino acid residues in proteins, using the genetic code. To read the genetic code, cells make a copy of a stretch of DNA in Ribonucleic acid (RNA) [19]. These RNA copies can then be used to direct protein synthesis [20] [21]. During the protein creation in higher organisms take a place a process of elimination of the superfluous DNA sequence. Points on a DNA sequence at which redundant DNA is removed calls splice junctions. The problem posed in this dataset is to recognize, given a sequence of DNA, the boundaries between exons (the parts of the DNA sequence retained after splicing) and introns (the parts of the DNA sequence that are spliced out). This problem consists of two subtasks: recognizing exon/intron boundaries (referred to as EI sites), and recognizing intron/exon boundaries (IE sites). (In the biological community, IE borders are referred to an "acceptors" while EI borders are referred to as "donors".) For our complexity estimation purpose we use molecular biology database titled as "Primate splice-junction gene sequences (DNA) with associated imperfect domain theory" donated by G. Towell, M. Noordewier, and J. Shavlik that is available in Machine Learning Repository of Bren School of Information and Computer Science University of California, Irvine (ftp site: ics.uci.edu) This benchmark data has the following main features:

All examples taken from Genbank 64.1 (ftp site: genbank.bio.net)

Number of Instances: 3190

Number of Attributes: 62

Missing Attribute Values: none

Class Distribution:

EI: 767 (25%)

IE: 768 (25%)

Neither: 1655 (50%)

We create on the *learning* phase file(s) that consist of the samples randomly chosen from database. The number of samples *m*, in this case is the amount of instances. Each instance has a category *c* and 60 sequential DNA nucleotide positions a_j $(0 < j < 61)$ and in this case the structure (a_{ij}, c_i) where *i* is the sample number which is sent to neurocomputer on the *learning*.

The second *classification* phase is identical to bench-mark testing. We generate *m* samples (instances) and their associated category, than compute the indicator-function Q_{i}.

Difference between those two bench-mark examples is that the probabilities of coincidence are different, because of different database size and classes' distributions. Moreover, the sequence of DNA encoded in a_j reflects a part of 3D (not 2D) space of a DNA double helix.

Tic-tac-toe endgame classification problem

The tic-tac-toe endgame dataset encodes the complete set of possible board configurations at the end of tic-tac-toe games, where "x" is assumed to have played first. The target concept is "win for x" (i.e., true when "x" has one of 8 possible ways to create a "three-in-a-row"). The dataset contains 958 instances without missing values, each with 9 attributes, corresponding to tic-tac-toe squares and taking on 1 of 3 possible values: "x", "o", and "empty".

From the view of data structure the tic-tac-toe endgame problem is similar to DNA patterns recognition, that's why technically we easy apply identical approach for *learning* and *classification*. Moreover, tic-tac-toe problem "seems to be" simpler from the data size view.

There are only 2 classes. Each array of attributes is shorter at least in 6 times. Every attribute has twice less variants of the value, plus it is 2D space. In next sections, especially section 6, we show delusiveness of such assumption.

4. APPROACH VALIDATION

To validate proposed approach we have conducted the experimentations with the following settings:

two IBM © ZISC-036 ® modes: LSUP/L1;

five different databases with increasing complexity;

eight variants of m value: 50, 100, 250, 500, 1000, 2500, 5000, 10000.

For each set of parameters, tests are repeated 10 times in order to get statistical average and to check the deviations of the tests Totally, 800 tests have been performed.

Fig. 4, Fig. 5, shows the charts of Q_i where i is the database index or pattern index. We expect that Q_5 for 10 sub-zones reaches a higher value than Q_1. Intuitively the problem corresponding to classification of 10 stripped sub-zones (Q_5) is more complex than for 2 (Q_1).

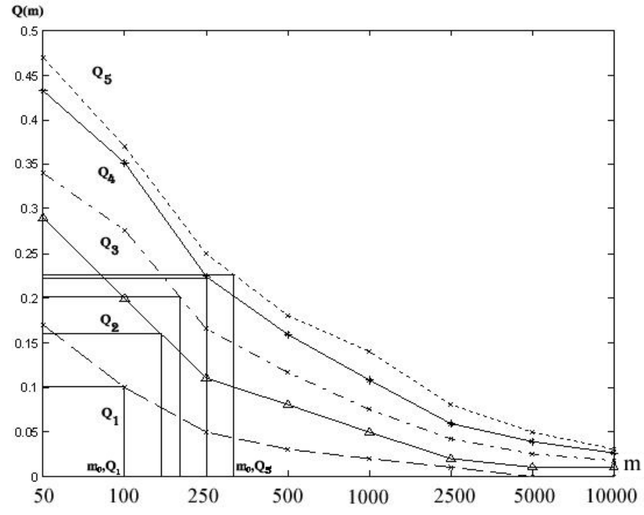


Fig. 4 – Coefficients of complexity Q_i(m) - LSUP ZISC-036 ® mode.

Consider a simple concept of view on the complexity of the learning process, in which a neurocomputer attempts to build complex neural structure in order to infer an unknown *classifying concept*, so regarding this process, *$Q_i(m)$ is a complexity learning curve.* The values of *$Q_i(m)$* are clearly closely related to each other. In either measuring we are interesting in the asymptotic behavior of a complexity of learning [22] as m becomes large. Since the curve can be used to determine how large m must be before the other parameters of classification such the rate of success reach a desire value.

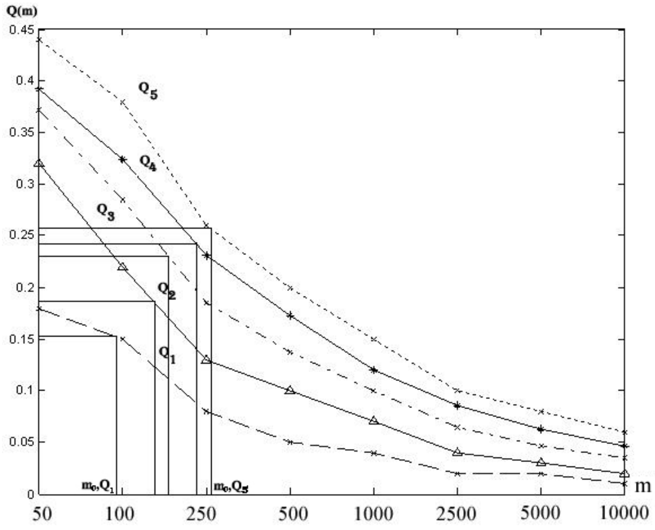


Fig. 5 – Coefficients of complexity Q_i(m) - L1 ZISC-036 ® mode.

The chart analysis suggests that exist(s) a point(s) m_j such as:

$$\frac{\partial^2}{\partial(m)}Q_i(m_j) = 0 \tag{3}$$

At such point m_j we have the following properties:

$$\forall m \geq 1, \exists \varepsilon_j > 0 : \forall m \in (m_j - \varepsilon_j ; m_j) \Rightarrow$$
$$\frac{\partial^2}{\partial(m)}Q_i(m_j) < 0, \forall m > m_j \Rightarrow \frac{\partial^2}{\partial(m)}Q_i(m_j) > 0 \tag{4.1}$$

or

$$\forall m \geq 1, \exists \varepsilon_j > 0 : \forall m \in (m_j - \varepsilon_j ; m_j) \Rightarrow$$
$$\frac{\partial^2}{\partial(m)}Q_i(m_j) > 0, \forall m > m_j \Rightarrow \frac{\partial^2}{\partial(m)}Q_i(m_j) < 0 \tag{4.2}$$

It means that there exists one or more point(s) m_j where the second derivative of Q_i changes its sign. Then we are interesting in m_0 defined by:

$$m_0 = \max(m_1,.., m_j,... m_k)$$
$$m_1 < .. < m_j < ... < m_k, m_0 = m_k \tag{5}$$

Where k is the number of points m_j. Main characteristic of the point m_0 is:

$$\forall m > m_0 : m \to +\infty \Rightarrow Q_i(m) \to const \tag{6}$$

In general case $const \geq 0$. The feature of the second derivative sign changes presents on the chart of *the rates of the success classification*, Fig. 6. That supports the idea of the strong influence second derivate has on the complexity estimation task. That fact turns a look on the problems not from the quantity side of complexity, but from the quality one.

It is clearly seen that in our specific bench-mark examples, the complexity of the classifying is lying in the range from Example 1 (2 zones, the easiest case) to Example 5 (10 zones, the complex one).

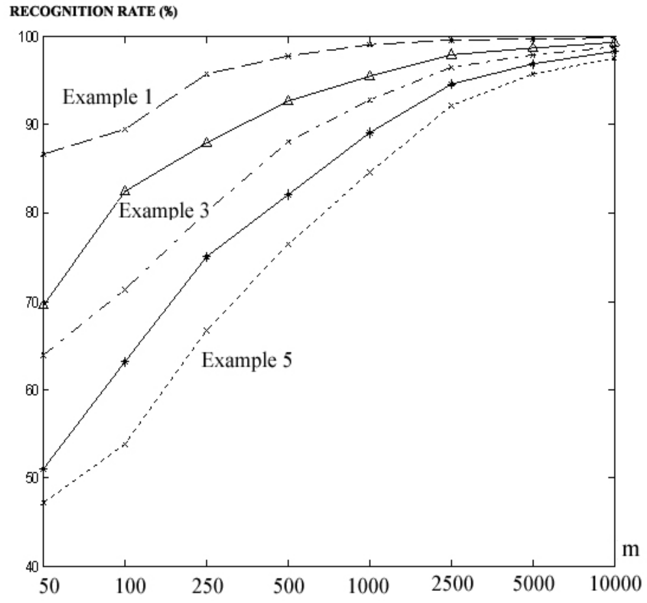


Fig. 6 – Rates of success of classification. Examples 1 – 5. LSUP ZISC-036 ® mode.

Analysis of the plots $m_{0,Q1}$ (Example 1) till $m_{0,Q5}$ (Example 5) for related classification tasks implies the following property:

$$m_{0,Q_1} < m_{0,Q_2} < m_{0,Q_3} < m_{0,Q_4} < m_{0,Q_5} \qquad (7)$$

In our particular tests we have (8), mentioning that in general (5), (7) and (8) are not obvious.

$$Q_1(m_0) < Q_2(m_0) < Q_3(m_0) < \\ < Q_4(m_0) < Q_5(m_0) \qquad (8)$$

On the other hand, we consider a particular value of *m* (an interesting value is m_0 for which the second derivative of $Q_i(m)$ changes the sign) stating $Q_i(m_0)$ *acting* as a *"complexity coefficient"*. In our case, $Q_i(m_0)$ acts as a critical checkpoint of classification process. The increase of m_0 stands for the classification task's complexity increasing.

Actually, many researchers try to identify general feature(s) of self organizing processes, so there is no revolutionary new in our approach. In particular [23], we point out that in many emergent and self-organizing processes, phase changes (from local to global behavior) occur at a well-defined critical value of some order parameter. For example, water freezes at a fixed temperature, nuclear chain reactions require a critical mass of fuel. So in next section we search the critical point(s) which in our approach represents its' complexity.

5. RESULTS

For DNA patterns recognition, we generate the 100 (amounts of the test) pairs (for *learning* and *classification* phases) of the files. For each set of the global parameter such as *m*, *ZISC-036 ® mode, etc.* appropriate pairs of the data files randomly have been generated from the given database in order to test, get good average parameters and to check the deviations. Approximately, up to 8400 tests have been performed.

After cubical polynomial approximation for 3 different initial modes we compute the coefficients of complexity $Q_i(m_0)$ Fig. 7.

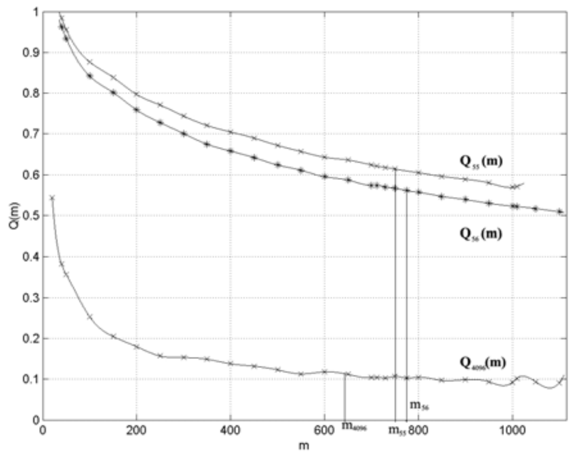


Fig. 7 – Evaluation of $Q_p(m)$ for DNA sequences recognition for different initializing parameter p, maximal influence field (MIF) of ZISC-036 ® : $Q_{55}(m)$, $Q_{56}(m)$, $Q_{4096}(m)$ modes.

The Table 1 represents the summary of the obtained chart results. Global optimizing parameter *MIF* is a *maximum influence field* used to initialize the RCE algorithm.

Table 1. Coefficient of complexity for DNA sequences recognition

Initial mode	m_0	$Q_i(m_0)$
MIF 55	730	0.618
MIF 56	775	0.561
MIF 4096	700	0.104

Fig. 8, Fig. 9, Fig. 10 supports the idea of the strong influence of the second derivate feature on the complexity estimation on the quality level of the recognitions.

For example on the Fig. 9, for calculated $m_{0,Q56}$ = *775* we can generally observe for *m* > $m_{0,Q56}$ *the rates of success classification* has a strong tendency to increase and *the rates of failure* – decrease. For *m* greater than critical $m_{0,Q55}$ *the rate of uncertainty* (the rate of the patterns, which cannot be classify by hardware tool, put to the special category) strongly concentrate around *14%*

The mentioned supports that $Q_p(m_0)$ **is the coefficient of the task complexity.**

Constructing cubical approximation for $Q_p(m)$ and calculating this coefficient for the best satisfying the initial ZISC-036 ® parameter is *MIF* = 56. The obtained *rate of the success classification* is 53.5%. *The rate of failure* – 15.6% are satisfied knowing

[21] that even using improved RBF methodology we cannot reach more than 66.3% of successful DNA pattern classifying [24]. That is why computed coefficient of complexity Q_{56} = 0.561, where $m_{0,Q56}$ = 775 Fig. 7, is acceptable and expected. Problem of DNA pattern classifying seems to be a priori complex.

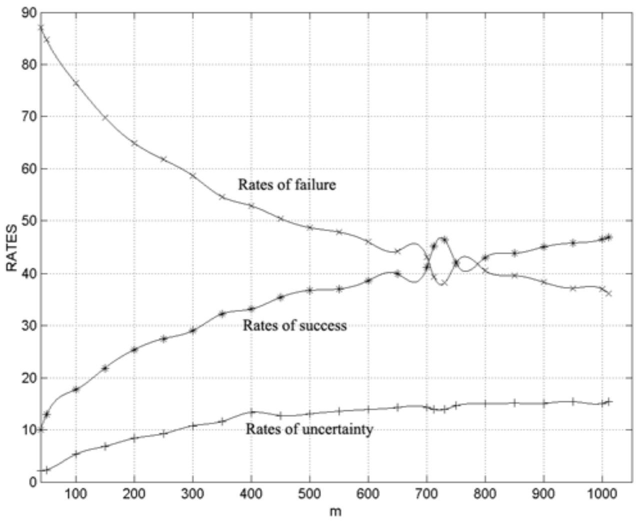


Fig. 8 – Rates of success, failure and uncertainty of classification for Q_{55}(m).

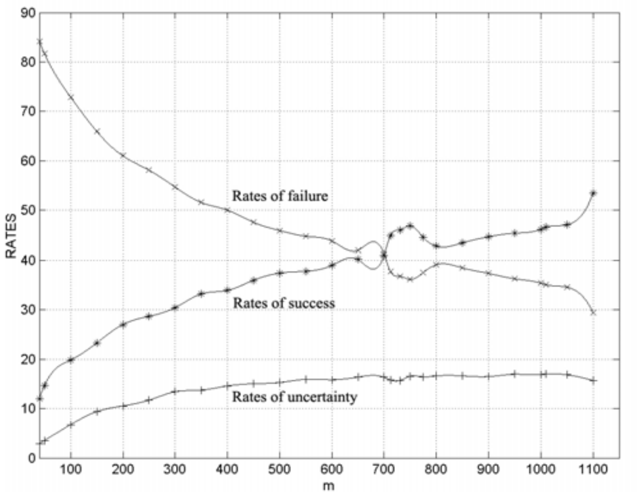


Fig. 9 – Rates of success, failure and uncertainty of classification for Q_{56}(m).

The same experimental protocol as for DNA benchmark is used for tic-tac-toe endgame problem. The datasets from the UCI machine learning repository is taken [25]. The database is randomly divided as in the reference sources into learning (90% of data) and recognition (remaining 10% of data) sets and to be comparable to the previous range of tests we also divide database on 50% of data for *learning* and 50% - for *classification*. The tests are applied to the training sets and this process is repeated 32 times for each data set (a pair of data files). We change during a testing a tuning initializing parameter *MIF*, so totally we performed around 1728 tests.

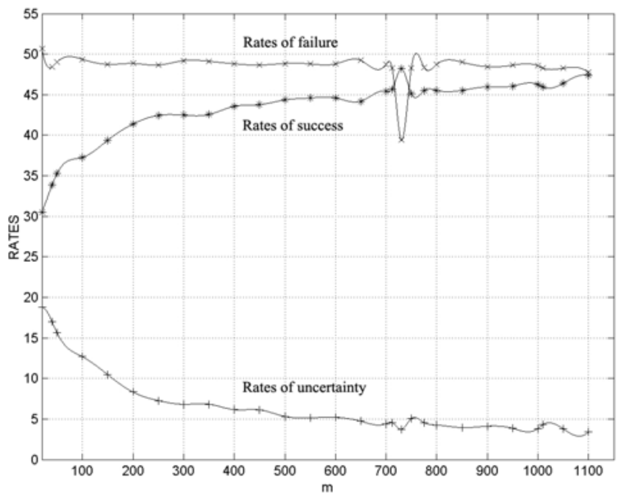


Fig. 10 – Rates of success, failure and uncertainty of classification for Q_{4096} (m).

The tic-tac-toe classification problem from the point of view of the covering type of algorithms is difficult [26] Fig. 11, as well as it is difficult for the methods based on the approach *divide and conquer* [27].

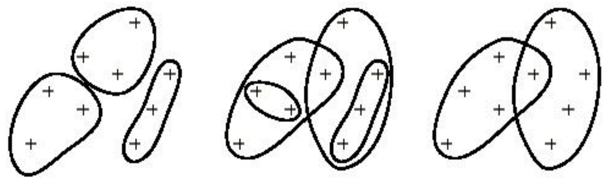


Fig. 11 – A covering family of algorithms applied for the tic-tac-toe classification problem.

Another group of methods, which use the idea of finding minimal description function and its' approximation for the class of the examples, are proven [28] to be NP-complete. Generally, the class of problems such as DNA patterns and tic-tac-toe endgame recognition are time and resource consuming and in some cases even impossible to examine all possible examples.

Table 2 contains the summary of the obtained chart results. Relatively optimal tuning factor of classification is *MIF* = 3. Even for this optimal initial parameter the rate of the successful classification reaches by default level 65% [26].

Table 2. Coefficient of complexity for tic-tac-toe endgame problem

Initial mode	m_0	$Q_i(m_0)$
MIF 3	545	0.6384
MIF 56	522	0.0839

Coefficient of complexity $Q_3(m_0)$ = 0.6384

In fact, tic-tac-toe problem for the proposed method of the classification is the most complex.

We can state so, because the other parameters as sample distribution (the main of them) are equal for DNA pattern classification and tic-tac-toe endgame problems.

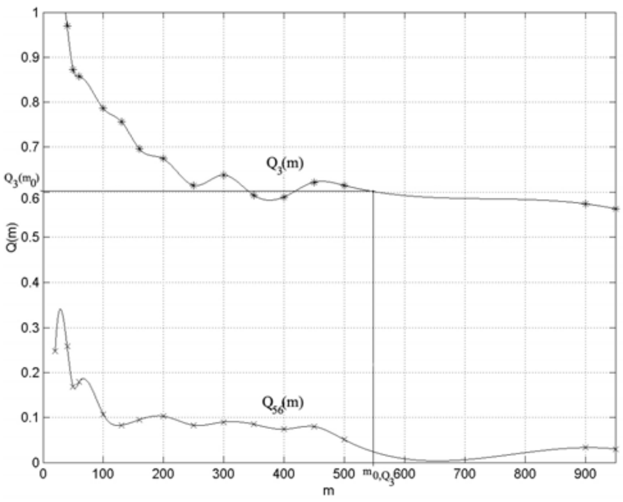


Fig. 12 – Evaluation of $Q_p(m)$ for tic-tac-toe endgame problem for different initializing parameter p, maximal influence field (MIF) of ZISC-036 ® : $Q_3(m)$, $Q_{56}(m)$, modes.

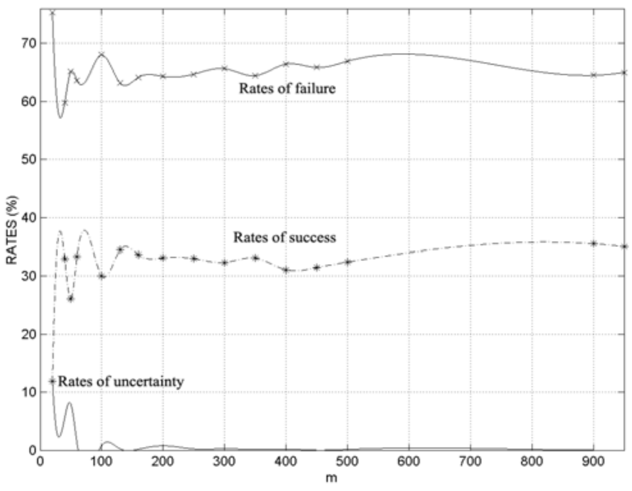


Fig. 13 – Rates of success, failure and uncertainty of classification for $Q_{56}(m)$.

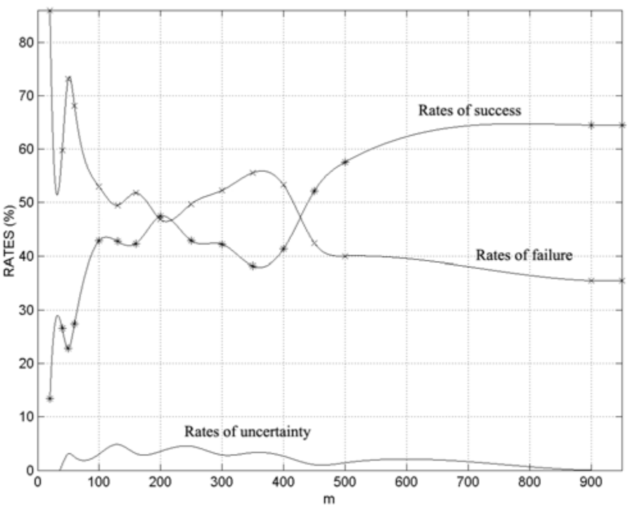


Fig. 14 – Rates of success, failure and uncertainty of classification for $Q_3(m)$.

The rates of the classification process for not optimal parameter *MIF = 56,* and optimal *MIF = 3* are shown on Fig. 13 and Fig. 14.

6. CONCLUSION

In this paper we describe a new method for complexity estimation based on an indicator-function related to neurocomputing technology.

The complexity indicator is extracted from some pertinent neural network structure parameter - the number of neurons in the structure.

The presented approach uses the following hypothesis: *more complex classification problem involves more complex neural network structures for its processing* relating to the initial condition where the other basic parameters are equal.

The presented concept has been implemented on IBM © ZISC-036 ® massively parallel neurocomputer and takes additional advantage of standard digital technology robustness and the high processing speed of this neuroprocessor.

It has been validated using a two-classes bench-mark set of classification paradigms with increasing complexity.

The proposed concept has been applied and verified using a three-class set of DNA patterns and two-class tic-tac-toe endgame classification problem.

Future research in this field will embed this approach in T-DTS in order to improve performance. A parallel stream of the development of this approach will be formalization *a complexity indicator* and specification of the other pertinent parameters to study their properties and their influence on the *integrated complexity coefficient*.

7. REFERENCES

[1] E. Bouyoucef. A. Chebira. M. Rybnik. K. Madani. Multiple Neural Network Model Generator with Complexity Estimation and self-Organization Abilities, *International Scientific Journal of Computing* (2005) Vol. 4. Issue 3, pp. 20–29.

[2] Laboratory IBM France. *ZISC® 036 Neurons User's Manual*. Version 1.2. Component Development (1998).

[3] K. Madani. M. Rybnik. A. Chebira. Data Driven Multiple Neural Network Models Generator Based on a Tree-like Scheduler. *Lecture Notes in Computer Science.* Edited J. Mira. A. Prieto. Springer Verlag (2003). ISBN 3-540-40210-1, pp. 382-389.

[4] M.A. DE Almeida. H. Lounis. W.L. Melo. An Investigation on the Use of Machine Learned Models for Estimating Software Correctability. *International Journal of Software Engineering and Knowledge Engineer (JSEE).* Volume 9. Issue 5. (October 1999), pp. 565-593.

[5] M. I. Jordan. L. Xu. Convergence *Results for the EM Approach to Mixture of Experts Architectures Neural Networks.* Pergamon. Elsevier (1995). Volume 8, N 9, pp. 1409-1431.

[6] K. Madani. A. Chebira. Data Analysis Approach Based on a Neural Networks Data Sets Decomposition and it's Hardware Implementation. *PKDD 2000*. Lyon, France 2000.

[7] G. DE Tremiolles. Contribution to the Theoretical Study of Neuro-Mimetic Models and to their Experimental Validation: a Panel of Industrial Applications. *Ph.D. Report*. University of Paris XII (1998).

[8] G. DE Tremiolles. P. Tannhof. B. Plougonven. C. Demarigny. K. Madani. Visual Probe Mark Inspection using Hardware Implementation of Artificial Neural Networks in VLSI Production. *Lecture Notes in Computer Science, Biological and Artificial Computation: From Neuroscience to Technology*. Edited by J. Mira. R. Diaz. M. J. Cabestany. Springer Verlag, Berlin. Heidelberg (1997) 1374-1383.

[9] Laboratory IBM France. *ISC/ISA Accelerator Card for PC*. User Manual. IBM France (1995).

[10] J. Wang. P. Neskovic. L. N. Cooper. Learning class regions by sphere covering. *IBNS Technical Report 2006-02*. March 2006. Department of Physics and Institute for Brain and Neural Systems Brown University. Providence. RI 02912.

[11] B. V. Dasarathy, editor (1991) *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques*. ISBN 0-8186-8930-7.

[12] J. Park. J. W. Sandberg. *Universal Approximation Using Radial Basis Functions Network, Neural Computation* (1991) Volume 3, pp. 246-257.

[13] J. Sima. R. Neruda. *Teoretické Otázky Neuronových Sítí*. MATFYZPRESS, Prague(1996), 390 pages.

[14] B. Edmonds. What is Complexity? - The philosophy of complexity per se with application to some examples in evolution. F. Heylighen & D. Aerts (Eds.) 1999. *The Evolution of Complexity*.

[15] A. Kohn. L. G. Nakano. V. Mani. A Class Discriminability Measure Based on Feature Space Partitioning. *Pattern Recognition* (1996) 29(5), pp. 873-887.

[16] S. F. Bush. On The Effectiveness of Kolmogorov Complexity Estimation to Discriminate Semantic Types, Senior member of IEEE, Todd Hughes, Ph.D.

[17] D. G. Green. D. Newth. Towards a Theory of Everything? – Grand Challenge in Complexity and Informatics. *Complexity international* (2000). Volume 8. ISSN 1320-0682.

[18] C. Lucas. *Quantifying Complexity Theory*. (2004).

[19] J. Watson. F. Crick. *Molecular Structure of Nucleic Acids, a Structure for Deoxyribose Nucleic Acid, Nature* (1953) 171 (4356), 737-8.

[20] J. M. Butler. *Forensic DNA Typing*, Elsevier (2001), pp. 14-15.

[21] A. Bruce. A. Johnson. J. Lewis. M. Raff, K. Roberts. P. Walters. *Molecular Biology of the Cell; Fourth Edition*. New York and London. Garland Science (2000).

[22] D. Haussler. M. Kearns. R. Schapire. Bounds on the Sample Complexity of Bayesian Learning Using Information Theory and the VC Dimension. *Proceeding of the 4th annual workshop on Computional Learning Theory.* Santa Cruz. Califonia. U.S. 1991, pp. 61-74. ISBN: 1-55860-213-5.

[23] H. Haken. *Synergetics.* Springer-Verlag, Berlin (1978).

[24] E. Bouyoucef, Comparaison des Performances de la T-DTS avec 34 Algorithmes de Classification en Exploitant 16 Bases de Données de l'UCI (Machine Learning Repository). *Ph.D. Report*. University of Paris XII (2007).

[25] D. W. Aha. Incremental Constructive Induction: An Instance-Based Approach. *In Proceedings of the Eight International Workshop on Machine Learning*. Morgan Kaufmann (1991), pp. 117-121.

[26] F. Torre. Contributions à l'apprentissage disjonctif. *Intégration des bias de langage à l'algorithme générer-et-tester*. Équipe Inférence et Apprentisage Laboratoire de Recherche en Informatique Université Paris-Sud. Rapport (28.02.2000), pp. 16-19.

[27] H. Bostrom. Covering vs. divide-and-conquer for top-down induction of logic programs. *Proceedings of the 14th International Joint Conference on Artificial Intelligence (1995)*, pp. 1194-1200.

[28] L. Dakovski. Z. Shevked. An Alternative Approach for Learning from Examples, International Conference on Computer Systems and Technologies – CompSysTech' 2005, pp. IIIB.5-1 – IIIB.5-6.

Ivan Budnyk *received his Specialist of Computer Science degree from the National University of "Kyiv-Mohyla Academy", in 2004. Since November 2006 he works as Ph.D. student in Images, Signals and Intelligent Systems Laboratory (LISSI / EA 3956) of PARIS XII – Val de Marne University. His research works deals with data processing, artificial learning, complexity estimation methods, classification techniques and parallel programming.*

Dr. Abdennasser Chebira *Received his Ph.D. degree in Electrical Engineering and Computer Sciences from PARIS XI University, Orsay, France, in 1994. Since September 1994 he works as Professor Assistant at Sénart Institute of Technology of PARIS XII – Val de Marne University. He is a staff researcher at Images, Signal and Intelligent Systems Laboratory (LISSI / EA 3956) of this University. His current research works concern self-organizing neural network based multi-modeling, hybrid neural based information processing systems; Neural based data fusion and complexity estimation.*

Prof. Kurosh Madani *received his Ph.D. degree in Electrical Engineering and Computer Sciences from University PARIS XI (PARIS-SUD), Orsay, France, in 1990. From 1989 to 1990, he worked as assistant professor at Institute of Fundamental Electronics of PARIS XI University. In 1990, he joined Creteil-Sénart Institute of Technology of University PARIS XII – Val de Marne, Lieusaint, France, where he worked from 1990 to 1998 as assistant professor. In 1995, he received the DHDR Doctor Habilitate degree (senior research Dr. Hab. degree) from University PARIS XII – Val de Marne. Since 1998 he works as Chair Professor in Electrical Engineering of Sénart Institute of Technology of University PARIS XII – Val de Marne. From 1992 to 2004 he has been head of Intelligence in Instrumentation and Systems Laboratory of PARIS XII – Val de Marne University located at Sénart Institute of Technology. Since 2005, he is head of one of the three research teams of Image, Signal and Intelligent Systems Laboratory (LISSI / EA 3956) of PARIS XII University. He has worked on both digital and analog implementation of processors arrays for image processing by stochastic relaxation, electro-optical random number generation, and both analog and digital Artificial Neural Networks (ANN) implementation. His current research interests include large ANN structures behavior modeling and implementation, hybrid neural based information processing systems and their software and hardware implementations, design and implementation of real-time neuro-control and neural based fault detection and diagnosis systems. Since 1996 he is a permanent member (elected Academician) of International Informatization Academy. In 1997, he was also elected as Academician of International Academy of Technological Cybernetics.*

computing@computingonline.net
www.computingonline.net

ISSN 1727-6209
International Scientific
Journal of Computing

ASSOCIATIVE MEMORIES NETWORK FOR FACE RECOGNITION AND OBJECT RECOGNITION

Roberto A. Vazquez [1], Humberto Sossa [2]

[1] Center for Computing Research (CIC-IPN), Av. Juan de Dios Batiz s/n Col. Nueva Industrial Vallejo, CP. 07738 Mexico City, Mexico, ravem@ipn.mx
[2] Center for Computing Research (CIC-IPN), Av. Juan de Dios Batiz s/n Colonia Nueva Industrial Vallejo CP. 07738 Mexico City, Mexico, hsossa@cic.ipn.mx

Abstract: *An associative memory* (AM) *is a special kind of neural network that allows associating an output pattern with an input pattern. Some problems require associating several output patterns with a unique pattern. Classical associative and neural models cannot solve this simple task and less if these patterns are complex images, for example faces. In this paper a network of AMs to recall a collection of patterns is proposed. The accuracy of the proposal is tested with two benchmarks. One is composed by 20 objects and the other is composed by 20 images of 15 different people faces. First the all, the benchmarks are split into several collections and then this collections are used to train the network of AMs. During training an image of a collection is associated with the rest of the images belonging to the same collection. Once trained the network we expected to recover a collection of images by using as an input pattern any image belonging to the collection.*

Keywords: *Associative memories, face recognition, object recognition.*

1. INTRODUCTION

An associative memory AM is a special kind of neural network that allows recalling one output pattern given an input pattern as a key that might be altered by some kind of noise (additive, subtractive or mixed). Several models of AMs are described in [1], [2], [3], [4], [5], [6], [7] and [8]. Most common application of these models is as a filter. In general, these models have several constraints which limit their applicability in complex problems such as object and face recognition. In particular, models described in [4], [5] and [6] cannot handle with mixed noise. Associative model presented in [7] and [8] is robust to mixed noise.

An association between input pattern $\mathbf{x}$ and output pattern $\mathbf{y}$ is denoted as $\left(\mathbf{x}^k,\mathbf{y}^k\right)$, where k is the corresponding association. AM $\mathbf{W}$ is represented by a matrix whose component w_{ij} can be seen as the synapse of the neural network. Operator $\mathbf{W}$ is generated from a finite a priori set of know associations, known as the fundamental set of association and is represented as: $\left\{\left(\mathbf{x}^k,\mathbf{y}^k\right)\middle|k=1,\ldots,p\right\}$ where p is the number of associations. If $\mathbf{x}^k=\mathbf{y}^k\forall k=1,\ldots,p$ then $\mathbf{W}$ is auto-associative, otherwise it is hetero-associative. A distorted version of a pattern $\mathbf{x}$ to be restored will be denoted as $\tilde{\mathbf{x}}$. If an AM $\mathbf{W}$ is fed with a distorted version of $\mathbf{x}^k$ and the output obtained is exactly $\mathbf{y}^k$, we say that recalling is perfect.

These models only associate an input pattern with a unique output pattern which in some problems could be seen as a limitation. Suppose for example that instead of recall a face of a person associated to an input pattern you want to recall different faces of the person, probably took in different stages of his life, using the same input pattern, or any pattern related to the face. In order to solve these kind problems, we will need more than one associative memory. We will need a network of associative memories because an associative memory only recalls an output pattern and not a collection of patterns.

In this paper we present how a network of AMs can be used to recall nor just one pattern but several of them given an input pattern. Further more we shown how the adopted associative model is also useful in complex problems such as object and face recognition. In this proposal an association between input pattern $\mathbf{x}$ and a collection of output pattern $\mathbf{Y}$ is denoted as, $\left\{\left(\mathbf{x}^k,\mathbf{Y}^k\right)\middle|k=1,\ldots,p\right\}$ where p is the number of association, $\mathbf{Y}^k=\left\{\mathbf{y}^1,\ldots,\mathbf{y}^r\right\}$ is a collection of output patterns and r is the number of patterns belonging to collection $\mathbf{Y}$.

The remaining of the paper is organized as follows. In sections 2 and 3 the associative model used in this research is described. In section 4 the proposed network of AMs is presented. In section 5 the experimental results obtained with the proposal are given. In section 6, finally, the conclusions and several directions for further research in this direction are presented.

2. DYNAMIC ASSOCIATIVE MODEL

The brain is not a huge fixed neural network, as had been previously thought, but a dynamic, changing neural network that adapts continuously to meet the demands of communication and computational needs [9]. This fact suggests that some connections of the brain could change in response to some input stimuli.

Humans, in general, do not have problems to recognize patterns even if these are altered by noise. Several parts of the brain interact together in the process of learning and recalling a pattern. For example, when we read a word the information enters the eye and the word is transformed into electrical impulses. Then electrical signals are passed through the brain to the *visual cortex*, where information about space, orientation, form and color is analyzed. After that, specific information about the patterns passes on the other areas of the *cortex* that integrate visual and auditory information. From here information passes through the *arcuate fasiculus*, a path that connects a large network of interacting brain areas; paths of this pathway connect language areas with other areas involving in cognition, association and meaning, for details see [10] and [11].

Based upon the above example we have defined in our model several interacting areas, one per association we would like the memory to learn. Also we have integrated the capability to adjust synapses in response to an input stimulus.

As we could appreciate from the previous example, before an input pattern is learned or processed by the brain, it is hypothesized that it is transformed and codified by the brain. In our model, this process is simulated using the following procedure recently introduced in [7]:

Procedure 1. Transform the fundamental set of associations into codified patterns and de-codifier patterns:

```
Input: FS Fundamental set of associations:
{1. Make $d = const$ and make $(\bar{\mathbf{x}}^1, \bar{\mathbf{y}}^1) = (\mathbf{x}^1, \mathbf{y}^1)$
 2. For the remaining couples do {
    For $k = 2$ to $p$ {
              For $i = 1$ to $n$ {
                  $\bar{x}_i^k = \bar{x}_i^{k-1} + d$ ;
    $€_i^k = \bar{x}_i^k - x_i^k$ ; $\bar{y}_i^k = \bar{y}_i^{k-1} + d$ ; $¥_i^k = \bar{y}_i^k - y_i^k$
 }}} Output: Set of codified and de-codifying patterns.
```

This procedure allows computing *codified patterns* from input and output patterns denoted by $\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}$ respectively; **€** and **¥** are *de-codifying patterns*. Codified and de-codifying patterns are allocated in different interacting areas and *d* defines of much these areas are separated. On the other hand, *d* determines the noise supported by our model. In addition a simplified version of $\mathbf{x}^k$ denoted by s_k is obtained as:

$$s_k = s\left(\mathbf{x}^k\right) = \mathbf{mid}\ \mathbf{x}^k \qquad (1)$$

where **mid** operator is defined as $\mathbf{mid}\ \mathbf{x} = x_{(n+1)/2}$.

When the brain is stimulated by an input pattern, some regions of the brain (interacting areas) are stimulated and synapses belonging to those regions are modified.

In our model, we call these regions *active regions* and could be estimated as follows:

$$ar = r\left(\mathbf{x}\right) = \arg\left(\min_{i=1}^{p}\left|s\left(\mathbf{x}\right) - s_i\right|\right) \qquad (2)$$

Once computed the *codified patterns*, the *de-codifying patterns* and s_k we can build the associative memory.

Let $\left\{\left(\bar{\mathbf{x}}^k, \bar{\mathbf{y}}^k\right) \middle| k = 1, \ldots, p\right\}, \bar{\mathbf{x}}^k \in \mathbf{R}^n, \bar{\mathbf{y}}^k \in \mathbf{R}^m$ a fundamental set of associations (codified patterns). Synapses of associative memory $\mathbf{W}$ are defined as:

$$w_{ij} = \bar{y}_i - \bar{x}_j \qquad (3)$$

After computed the *codified patterns*, the *de-codifying patterns,* the reader can easily corroborate that any association can be used to compute the synapses of $\mathbf{W}$ without modifying the results. In short, building of the associative memory can be performed in three stages as:

1. Transform the fundamental set of association into codified and de-codifying patterns by means of previously described Procedure 1.
2. Compute simplified versions of input patterns by using equation 1.
3. Build $\mathbf{W}$ in terms of codified patterns by using equation 3.

As we had already mentioned, synapses could change in response to an input stimulus; but which synapses should be modified? For example, a head injury might cause a brain lesion killing hundred of neurons; this entails some synapses to reconnect with others neurons. This reconnection or modification of the synapses might cause that information allocated on brain will be preserved or will be lost, the reader could find more details concerning to this topic in [12] and [13].

This fact suggests there are synapses that can be drastically modified and they do not alter the behavior of the associative memory. In the contrary, there are synapses that only can be slightly modified to do not alter the behavior of the associative memory; we call this set of synapses *the kernel* of the associative memory and it is denoted by $\mathbf{K}_\mathbf{W}$.

In the model we can find two types of synapses: synapses that can be modified and do not alter the behavior of the associative memory; and synapses belonging to the kernel of the associative memory. These last synapses play an important role in recalling patterns altered by some kind of noise.

Let $\mathbf{K}_\mathbf{W} \in \mathbf{R}^n$ the kernel of an associative memory $\mathbf{W}$. A component of vector $\mathbf{K}_\mathbf{W}$ is defined as:

$$kw_i = \mathbf{mid}\left(w_{ij}\right), j = 1, \ldots, m \tag{4}$$

According to the original idea of our proposal, synapses that belong to $\mathbf{K}_\mathbf{W}$ are modified as a response to an input stimulus. Input patterns stimulate some *active regions*, interact with these regions and then, according to those interactions, the corresponding synapses are modified. Synapses belonging to $\mathbf{K}_\mathbf{W}$ are modified according to the stimulus generated by the input pattern. This adjusting factor is denoted by Δw and can be computed as:

$$\Delta w = \Delta\left(\mathbf{x}\right) = s\left(\overline{\mathbf{x}}^{ar}\right) - s\left(\mathbf{x}\right) \tag{5}$$

where ar is the index of the *active region*.

Finally, synapses belonging to $\mathbf{K}_\mathbf{W}$ are modified as:

$$\mathbf{K}_\mathbf{W} = \mathbf{K}_\mathbf{W} \oplus \left(\Delta w - \Delta w_{old}\right) \tag{6}$$

where operator $\oplus$ is defined as $\mathbf{x} \oplus e = x_i + e \ \forall i = 1, \ldots, m$. As you can appreciate, modification of $\mathbf{K}_\mathbf{W}$ in equation 6 depends of the previous value of Δw denoted by Δw_{old} obtained with the previous input pattern. Once trained the **AM**, when it is used by first time, the value of Δw_{old} is set to zero.

Once synapses of the associative memory have been modified in response to an input pattern, every component of vector $\overline{\mathbf{y}}$ can be recalled by using its corresponding input vector $\overline{\mathbf{x}}$ as:

$$\overline{y}_i = \mathbf{mid}\left(w_{ij} + \overline{x}_j\right), j = 1, \ldots, n \tag{7}$$

In short, pattern $\overline{\mathbf{y}}$ can be recalled by using its corresponding key vector $\overline{\mathbf{x}}$ or $\tilde{\mathbf{x}}$ in six stages as follows:

1. Obtain index of the active region ar by using equation 2.
2. Transform $\mathbf{x}^k$ using de-codifying pattern $\overline{\mathbf{x}}^{ar}$ by applying the following transformation: $\widehat{\mathbf{x}}^k = \mathbf{x}^k + \overline{\mathbf{x}}^{ar}$.
3. Compute adjust factor $\Delta w = \Delta\left(\widehat{\mathbf{x}}\right)$ by using equation 5.
4. Modify synapses of associative memory $\mathbf{W}$ that belong to $\mathbf{K}_\mathbf{W}$ by using equation 6.
5. Recall pattern $\widehat{\mathbf{y}}^k$ by using equation 7.
6. Obtain $\mathbf{y}^k$ by transforming $\widehat{\mathbf{y}}^k$ using de-codifying pattern $\overline{\mathbf{y}}^{ar}$ by applying transformation: $\mathbf{y}^k = \widehat{\mathbf{y}}^k - \overline{\mathbf{y}}^{ar}$.

The formal set of prepositions that support the correct functioning of this dynamic model can be found in [14].

3. MODIFIED DAM

In [15] the authors describe an interesting idea for recognizing faces based on the random selection of stimulating points (pixels), see figure 1.

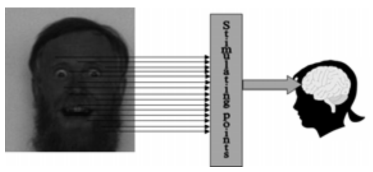


Fig. 1 – Random selection of stimulation points used to train an AM.

In order to recognize complex images such as faces, we add to the DAM model a vector of stimulating points $\mathbf{SP}$ where each stimulating point, given by $sp_i = random\left(n\right)$, is a random number between zero and the length of input pattern and $i = 1, \ldots, c$ where c is the number of stimulating points used. To determine the active region we allocate in the DAM model an alternative simplified version of each pattern $\mathbf{x}^k$ given by:

$$\mathbf{ss}_i^k = ss\left(\mathbf{x}^k\right) = \mathbf{x}_{sp_i}^k \tag{8}$$

Once compute these simplified versions we could estimate the active region as follows:

$$ar = r(\mathbf{x}) = \arg\max_{i=1}^{p}(\mathbf{a}) \tag{9}$$

where $a_{b_i} = a_{b_i} + 1$, $b_i = \arg\min_{k=1}^{p} \left| \left[ss(\mathbf{x}) \right]_i - \mathbf{ss}_i^k \right|$ and $i = 1, \ldots, c$.

Building phase of the DAM is done as follows:

1. Let $\mathbf{I}_x^k$ and $\mathbf{I}_y^k$ an association of images and c be the number of stimulating points.
2. Take at random a stimulating point $sp_i, i = 1, \ldots, c$.
3. For each association:
4. Select filter size and apply it to the stimulating points in the images.
5. Transform the images into a vector $(\mathbf{x}^k, \mathbf{y}^k)$ by means of the standard image scan method.
6. Train the DAM as in building procedure and compute the alternative simplified version of the patterns by using equation 8.

Pattern $\mathbf{I}_y^k$ can be recalled by using its corresponding key image $\mathbf{I}_x^k$ or distorted version $\tilde{\mathbf{I}}_x^k$ as follows:

1. Use the same stimulating point, $sp_i, i = 1, \ldots, c$ and filter size as in building phase.
2. Apply filter to the stimulating points in the images.
3. Transform the images into a vector by means of the standard image scan method
4. Determine active region using equation 9.
5. Apply steps from two to six as described in recalling procedure.

4. ARCHITECTURE OF THE NETWORK

Classical AMs (see for example [1], [2], [3], [4], [5], [6], [7] and [8]) are able to recover a pattern (an image) from a noisy version of it. In their original form classical AMs are not useful when image is altered by image transformations, such as translations, rotations, and so on.

The network of AMs proposed in this paper is robust under some of these transformations. Taking advantage of this fact, we can associate different versions of an image (rotated, translated and deformed) to an image.

Our task is to propose a network of AMs aimed to associate an image with other images belonging to the same collection. In order to achieve this, first suppose we want to associate images belonging to a collection with an image of the same collection using an AM. A good solution could be to compute the average image of whole images belonging to the collection and then associate the average image with any image that belongs to the collection. The same solution can be applied to other collections. Once computed the average images from different collections and chosen the images to be associated, we can train the AM as was described in section 2.

Until this point the AM only can recover an association between a collection of input patterns $\mathbf{X}$ and output pattern $\mathbf{y}$ denoted as, $\left\{ \left(\mathbf{X}^k, \mathbf{y}^k \right) \middle| k = 1, \ldots, p \right\}$ where p is the number of association, $\mathbf{X}^k = \left\{ \mathbf{x}^1, \ldots, \mathbf{x}^r \right\}$ is a collection of input patterns and r is the number of patterns belonging to collection $\mathbf{X}$. This means that it can only be recovered the associated image using any image from a collection. However, we would like to get the inverse result; instead of recovering the associated image using any image from a collection, we would like to recover all the images belonging to the collection using any image of the collection.

To achieve this goal we will train a network of AMs built as in previous sections. Each AM will associate all the images of a collection with one image of this collection. This implies that for recovering all images of a given collection, we would need r AMs, where r is the number of images belonging to the collection. The network architecture of AMs needed for recovering a collection of images is shown in Fig. 2.

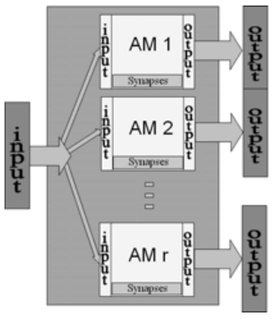


Fig. 2 – Architecture of a network of AMs for recalling a collection of patters using an input pattern.

In order to train the network of r AMs, first of all we need to know the number of collections we want to recover. Training phase is done as follows:

1. Transform each image into a vector.
2. Build n collections of images $\left[\mathbf{CI}^n \right]_{q \times r}$ where q is the number of pixels of each image and r the number of images.
3. Let $\left[\mathbf{AI} \right]_{q \times n}$ a matrix of average images. For $k = 1$ to n compute the average image as:
$$\mathbf{AI}_k = \frac{\sum_{s=1}^{r} \mathbf{CI}_s^k}{r} \tag{10}$$
4. For $s = 1$ to r build an $\mathbf{AM}^s$ as described in Section 4 (building phase). For $k = 1$ to n

$\mathbf{x}^k = \mathbf{AI}_k$ and $\mathbf{y}^k = \mathbf{CI}_s^k$

Once trained the network of AMs, when is fed with any image of a collection, each AM will respond with an image that belongs to the collection. To recover a collection of images we just operate each AM with the input image as described in Section 3 (recalling phase).

Some important to remark is that once the collection of associative memories has been stimulated by an input pattern, we do not need any decision rule to determine the final class. Each associative memory will recall the corresponding output pattern which belongs to the same collection.

6. EXPERIMENTAL RESULTS

In this section the accuracy of the proposal is tested using two different benchmarks of images, see Fig. 3 and Fig. 4.

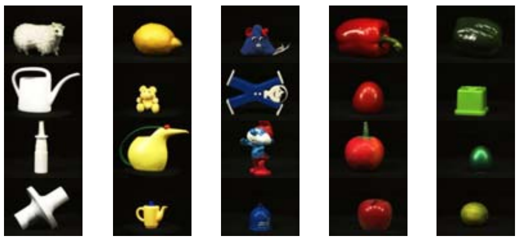


Fig. 3 – (a-e) Collections of images taken from the Amsterdam Library of Objects Images (ALOI).

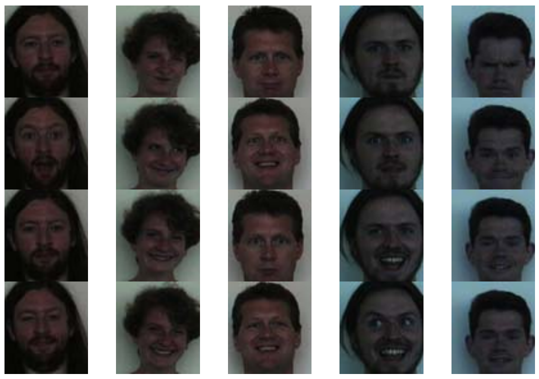


Fig. 4. (a-e) Collections of images taken from the Essex Collection of facial images.

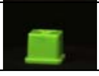
From the first benchmark 20 complex images were grouped into 5 collections composed by 4 images. After that, we proceeded to train the network of AMs as was explained in Section 5. For this problem our network is composed of four AMs. It is important to say that the number of images composing a collection could by any, the only restrictions to guaranty perfect and robust recall is that patterns (images) satisfy propositions described in [14].

Once trained the network, five experiments were performed to test the accuracy of the proposal. The first four experiments use the original DAM and the last experiment uses the modified DAM. The first experiment consisted on recovering a collection of images using any image of the collection in order to verify how much robust is the proposal under image deformations. Second experiment consisted on recovering a collection of images using any image of the collection altered by mixed noise in order to verify how much robust is the proposal under deformations and noisy version of the images. In the third experiment each image of the training set was rotated (from 0 to 360). We then used them to fed the network of AMs in order to verify how much robust is the proposal under deformations and rotations. Finally for forth and five experiments the images of the training set were rotated (from 0 to 360) and translated, and then used them to fed the network of AMs in order to verify how much robust is the proposal under deformations, rotations and translations.

The accuracy of the proposal was of 100% in the first experiment. The five collections of images where perfectly recovered by using any image of the collection (20 images), in Table 1 are shown some results obtained in this experiment. Remember that we train the network of AMs with average images, so then; when we fed the network with an image of any collection this image could be seen as a deformed version of the average images.

The results provided by our proposal in this experiment show that the associative model used to train the network of AMs are robust under deformations. Something important to say is that if we use other associative models for training the network of AMs such as morphological or median AMs, the collections might not be correctly recovered due to they are not robust under this kind of transformations or deformations.

Table 1. Some results obtained for the first experiment. As you can appreciate all sets of images were perfectly recovered.

Input image	Image recovered by each AM.			
				

The accuracy of the proposal, in the second experiment, was of 100%. The five collections of images where recovered perfectly by using any image of the collection as input, even when these images were altered by noise (200 images). As you can see in Table 2, despite of the level of noise added to the images, the collections were correctly recovered. Despite of other associative models are robust to this kind of noise, they might not recover the all collections due to they are only robust under additive, subtractive and mixed noise but not to image transformations.

The accuracy of the proposal, in the third experiment, was also of 100%. The five collections of images where recovered perfectly even when rotated version of the images were used (700 images), see in Table 3. Some important to say is, to our knowledge, neither morphological AMs nor other classical models are robust under rotations. Due to we used simplified patterns using mid operator and due to this operator is invariant to rotations the accuracy of the proposal was of 100%.

Table 2. Some results obtained for the second experiment. Despite of the noise added to the images, all sets of images were again correctly recovered.

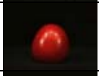
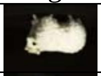
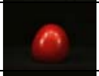
Input image	Image recovered by each AM			
				

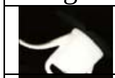
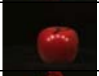
Table 3. Some results obtained for the third experiment. Despite of the noise added to the images and rotations, all sets of images were again correctly recovered.

Input image	Image recovered by each AM			
				
				

The accuracy of the proposal, in the third experiment, was also of 100%. The five collections of images where recovered perfectly even when rotated version of the images were used (700 images), see in Table 3. Some important to say is, to our knowledge, neither morphological AMs nor other classical models are robust under rotations. Due to we used simplified patterns using **mid** operator and due to this operator is invariant to rotations the accuracy of the proposal was of 100%.

The accuracy of the proposal, in the fourth experiment was of 40%. In this experiment with 700 images; with some images we recalled a collection, but with some other images (when patterns did not satisfied the proposition that guarantees robust recall) the collections were not recalled, see Table 4. However, the results obtained by our proposal are acceptable if they are compared with the results provided by order associative models (less of 10% of accuracy).

Table 4. Some results obtained for the fourth experiment. With some images we recalled a collection, but with some other images (when patters did not satisfied the proposition that guarantees robust recall) the collections were not recalled.

Input image	Image recovered by each AM			
				
				

Finally, if we use the modified DAM the accuracy increases, in average, to more than 80%. In this experiment we selected different numbers of stimulation points, as in [15].

For the second benchmark 300 complex images of faces were grouped into 15 collections composed by 20 images. After that, we proceeded to train the network of AMs as was explained in Section 5. For this problem our network was composed of 15 AMs.

Once trained the network, two experiments were performed to test the accuracy of the proposal. The first experiment uses the original DAM and the second experiment uses the modified DAM. The first experiment consisted on recovering a collection of faces using any face of the collection in order to verify how much robust is the proposal under face recognition. In these experiments we did not include noise or affine transformation because of the different gesticulations in the faces.

The accuracy of the proposal using the original DAM was of 20%. In this experiment with 300 images; with some images we recalled a collection, but with some other images (when patterns did not satisfy the proposition that guarantees robust recall), due to the gesticulations and illumination changes, the collections were not recalled.

For the last experiment, using the modified DAM, the accuracy increased to more than 95%. These results were obtained with more than 100 stimulating points.

In general, the accuracy of the proposal with different banks of images (altered by noisy and rotated) was of 100%. This was due to the input patterns (the images) satisfy the propositions presented in [14]. If these patterns do not satisfy these propositions, as images used in experiment four (translated and rotated images), the accuracy of the proposal diminish. However, the results provided by our proposal using the modified DAM, up-performed the results provide by the original DAM and other associative models.

5. CONCLUSION

In this paper we have proposed a network of AMs. This network is useful for recalling a collection of output patterns using an input pattern as a key. The network is composed by several dynamic associative memories (DAM). This DAM is inspired in some aspects of human brain. The model, due to plasticity of its synapses and functioning, is robust under some transformations as rotation, translation and deformations.

In addition, we describe an algorithm for training a network of AMs codifying the images of a collection by using an average image. Once computed the average images we proceed to training the network of AMs.

The network is capable to recall a collection of images (patterns) even if images are altered by noise or suffer some deformations and rotations.

Due to the original DAM present some problems with translations and face recognition we modified the DAM using a random selection technique. This technique allow the DAM up-perform the results obtained with the original DAM.

Through several experiments we have shown the efficiency of the proposal. In the first three experiments the proposal provided an accuracy of 100%. Even when the images were altered with mixed noise and rotated, the network of AMs recovered the corresponding collection. When object in images suffer translations or the object are faces, the accuracy of the proposal diminished. This is because most of the patterns under this transformation do not satisfied the propositions that guarantee robust recall. By using the modified DAM the accuracy of the proposal increases to more than 80% with translated objects and almost the 100% with complex faces. In general, the results provided by our proposal up-performed the results provide by other associative models.

The performed experiments could be seen as an application in image retrieval problems. We could say that we have developed a small system able to recover a collection of images (previously organized), even in the presence of altered versions of the images.

Nowadays we are working and directed this research to solve real problems related with signal and image retrieval. We are focusing our efforts to propose new associative models able to associate and recall images under more complex transformations. Furthermore, this new models have to work with images of much more complicated objects such as flowers, animals, cars, etc.

ACKNOWLEDGMENTS

This work was economically supported by SIP-IPN under grants 20071438 and 20082948 and CONACYT under grant 46805.

6. REFERENCES

[1] K. Steinbuch. 1961. Die Lernmatrix. *Kybernetik,* 1(1):26-45.

[2] J. A. Anderson. 1972. A simple neural network generating an interactive memory. *Mathematical Bioscien*ces, 14:197-220.

[3] J. J. Hopfield. 1982. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences,* 79: 2554-2558.

[4] G. X. Ritter el al. 1998. Morphological associative memories. *IEEE Transactions on Neural Network*s, 9:281-293.

[5] P. Sussner. 2003. Generalizing operations of binary auto-associative morphological memories using fuzzy set theory. *Journal of Mathematical Imaging and Vision,* 19(2):81-93.

[6] G. X. Ritter, G. Urcid, L. Iancu. 2003. Reconstruction of patterns from noisy inputs using morphological associative memories. *J. of Math. Imaging and Vision,* 19(2):95-111.

[7] H. Sossa, R. Barrón, R. A. Vázquez. 2004. Transforming Fundamental set of Patterns to a Canonical Form to Improve Pattern Recall. *LNAI,* 3315:687-696.

[8] H. Sossa, R. Barrón, R. A. Vázquez. 2004. New associative memories for recall real-

valued patterns. *LNCS*, 3287:195-202.

[9] S. B. Laughlin and T. J. Sejnowski. 2003. Communication in neuronal networks. *Science*, 301:1870-1874.

[10] M. Kutas and S. A. Hillyard. 1984. Brain potentials during reading reflect word expectancy and semantic association. *Nature*, 307:161–163.

[11] C. J. Price. 2000. The anatomy of language: contributions from functional neuroimaging. *Journal of Anatomy*, 197(3): 335-359.

[12] I. Reinvan. 1998. Amnestic disorders and their role in cognitive theory. *Scandinavian Journal of Psychology*, 39(3): 141-143. D. Jovanova-Nesic and B. D. Jankovic. 2005. The Neuronal and Immune Memory Systems as Supervisors of Neural Plasticity and Aging of the Brain. *Annals of the New York Academy of Sciences*, 1057: 279-295.

[13] R. A. Vázquez and H. Sossa. 2007. A new associative memory with dynamical synapses. (Submitted to Neural Processing Letters, 2008).

[14] R. A. Vázquez, H. Sossa and B. Garro. 2007. 3D object recognition based on low frequency responses and random feature selection. To appear in *LNAI* vol. 4827.

***Roberto A. Vazquez**, was born in Mexico City, Mexico, 1981. He received his B. Sc. degree from the School of Computer Sciences, National Polytechnic Institute (ESCOM-IPN), Mexico City, Mexico, 2003. He received his M. Sc. degree from the Center for Computing Research, National Polytechnic Institute (CIC-IPN), Mexico City, Mexico, 2005.*

He is now pursuing the Ph. D. degree at the Center for Computing Research, National Polytechnic Institute (CIC-IPN), Mexico City, Mexico.

His main research interests are Artificial Intelligence, Neurocomputing, Computational Neuroscience, Associative Memories, Pattern Recognition and Image Analysis.

***Humberto Sossa**, was born in Guadalajara, Jalisco, Mexico in 1956. He received a B. Sc. Degree in Electronics from the University of Guadalajara in 1981, a M.Sc. in Electrical Engineering from CINVESTAV-IPN in 1987 and a Ph.D. in Informatics from the National Polytechnic Institute of Grenoble, France in 1992.*

He is Full Time Professor at the Center for Computing Research of the National Polytechnic Institute of Mexico.

His main research interests are in Pattern Recognition, Neurocomputing, Computational Neuroscience, Associative Memories, Image Analysis, and Robot Control using Image Analysis.

computing@computingonline.net
www.computingonline.net

ISSN 1727-6209
International Journal
of Computing

FACE DETECTION ON GRAYSCALE AND COLOR IMAGES USING COMBINED CASCADE OF CLASSIFIERS

Yuriy Kurylyak [1], Ihor Paliy [1], Anatoly Sachenko [1], Amine Chohra [2], Kurosh Madani [2]

[1] Research Institute of Intelligent Computer Systems,
Ternopil National Economic University,
3 Peremoga Square, 46004, Ternopil, Ukraine
{yuk, ipl, as}@tneu.edu.ua
[2] Images, Signals and Intelligent Systems Laboratory (LISSI / EA 3956),
PARIS XII University, Senart-FB Institute of Technology
Av. Pierre Point, Bat. A, F-77127, Lieusaint, France
{chohra, madani}@univ-paris 12.fr

Abstract. *The paper describes improved face detection methods for grayscale and color images using the combined cascade of classifiers and skin color segmentation. The combined cascade with proposed face candidates' verification method allows achieving one of the best detection rates on CMU test set and a high processing speed suitable for a video flow processing. It's also shown that the mixture of color spaces is more efficient during the skin color segmentation than the application of one color space. A lot of experiments are made to choose rational parameters for the developed face detection system in order to improve the detection rate, false positives' number and system's speed.*

Keywords: *Face Detection, Skin Color Segmentation, Haar-like Features' Cascade of Weak Classifiers, Convolutional Neural Network, Combined Cascade of Classifiers.*

1. INTRODUCTION

During the last decade the researches in the face detection (FD) area became more and more active. The reason is that there are a lot of applications of this task such as face recognition, video-conferences, content-based image retrieval, video surveillance, human-computer interface, face expressions' analysis, visitors' counting, access control where the detection is the first stage of any face processing [1]. The FD task is quite easy for the humans but becomes a serious problem while developing an automatic detection system. In this case it is necessary to deal with a lot of factors which affect the face appearance such as viewpoint, slope, face expression, occlusions, light conditions, etc.

There are number of existing FD approaches but in respect to the detection rate and false positives' number (FP) more effective are the methods from appearance-based group [1]. Some of them like neural networks [2, 3, 4, 5], support vector machines [6, 7, 8], Bayesian classifier [9] make use of the monolithic classifiers and some of them are based on the cascade of classifiers [10, 11, 12, 13, 14, 15].

In general, the FD methods based on the face modeling show good results on the complex test sets (for example, on CMU test set [3] which includes 130 images with 507 faces) and are able to process more than 10 images per second. There is also the range of applications where the existent FD methods are not suitable, for example, video surveillance systems and access control. In such systems to achieve the high detection rate with the lowest false positive detections it is necessary to use the complex monolithic classifiers. However, their usage with the existing face search strategies lowers the performance of FD subsystem and makes impossible processing of the video stream. Methods with the cascade of classifiers allow fast video information processing but are not so effective. For example, with increasing of the detection rate by more than 90% the false positive detection rate increases exponentially [12]. Thus, the development of new FD methods that will show a high face detection rate and low number of false positive detections is very actual task. The important point also is the possibility to work with the video stream.

In this paper we research the proposed in [16, 17] FD methods for effective processing of grayscale and color images by applying the skin color segmentation and combined cascade of classifiers.

Moreover, in order to configure the rational parameters of developed FD system we provide the experimental results of such researches in respect to the detection rate, number of false positive detections as well as the performance.

2. FACE DETECTION ON GRAYSCALE IMAGES

The cascade structure of the FD classifier allows achieving high image processing speed due to the fast background rejection and paying more attention to the face-like regions. But in comparison with the monolithic classifiers the cascade classifier (for example, Haar-like features' cascade of weak classifiers [10]) increases the detection error and FP rate. The extension of Haar-like features' set [12] as well as improvement of the training algorithm [11] for the cascade of weak classifiers allows increasing the detection rate only by 7-8% for the low FP rate. Therefore, to achieve higher detection and lower FP rates it is necessary to join the quick cascade classifier and accurate monolithic one within the two-level combined cascade of classifiers instead of using them independently. The two-level cascade of classifiers is called "combined" because it combines the different-nature classifiers, which are chosen and justified by authors: the first level is represented by the Haar-like features' cascade of weak classifiers, which is responsible for the face candidates' detection, and the second level is a convolutional neural network for the candidates' verification (Fig. 1). We should also mention that there is no input image preprocessing stage as well as in [5].

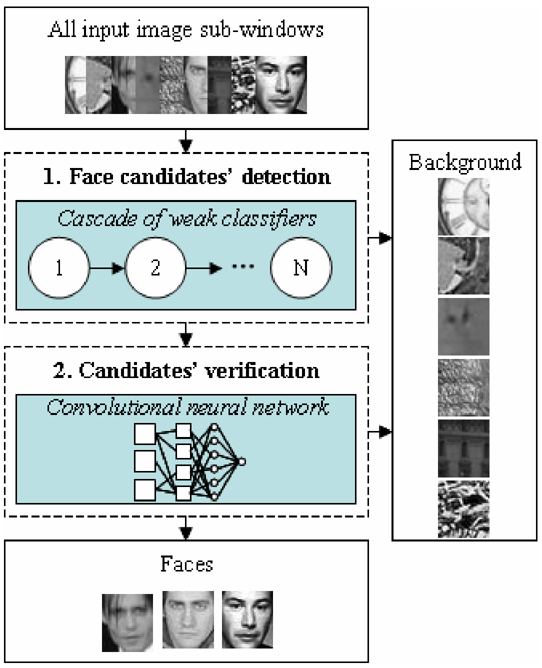

All input image sub-windows
1. Face candidates' detection
Cascade of weak classifiers
1
2
...
N
2. Candidates' verification
Convolutional neural network
Background
Faces

Fig. 1 – Face detection process using combined cascade of classifiers.

The Haar-like features' cascade of weak classifiers (CWC) [10] allows detecting face candidates very quickly (near real-time mode for the video flow). The CWC consists of levels which include one or more weak classifiers. The weak classifier's input is represented by Haar-like feature of the rectangular form which is composed of "white" and "black" rectangles. The feature's value is [10]

$$f(x) = w_w \times S_w + w_b \times S_b,$$

where x – input image sub-window, w_w and w_b – whole rectangle's and its black part's weights accordingly, S_w and S_b – whole rectangle's and its black part's sums of pixels (these sums are calculated very fast using an integral image [10]). A weak classifier's output value defines as

$$h(x) = \begin{cases} 1, \text{if } p \times f(x) < p \times \theta \\ -1, \text{otherwise} \end{cases},$$

where p – polarity, which indicates the direction of the inequality, θ – weak classifier's threshold. The cascade of weak classifiers is a linear combination of weak classifiers which is evaluated as [10]

$$H(x) = \sum_{t=1}^{T} w_t \times h_t(x),$$

where T – weak classifiers' number, w_t – t-weak classifier's weight. The AdaBoost algorithm [18] is used for the CWC's training and the selection of the most important Haar-like features.

The verification level uses the convolutional neural network (CNN) [5] which is more robust to the input images' deformations (shifts, scales, rotations, occlusions) in the classification issues due to the architectural features than other known classifiers [19]. The output value of a neuron with the coordinates (m,n) of p-plane and l-layer is [16]

$$y_{m,n}^{l,p}(x) = F(S_{m,n}^{l,p}(x)), \qquad (1)$$

where x – input face candidate's image, F – neuron's transfer function, $S_{m,n}^{l,p}(x)$ – neuron's weighted sum. According to the recommendation in [20] the bipolar sigmoid transfer function is used for CNN with the output range [-1; 1]. Therefore, the expression (1) may be represented as

$$y_{m,n}^{l,p}(x) = \frac{2}{1+\exp(-S_{m,n}^{l,p}(x))} - 1,$$

where a neuron's weighted sum is:

$$S_{m,n}^{l,p}(x) = (\sum_{k=0}^{K-1}\sum_{r=0}^{R-1}\sum_{c=0}^{C-1} y_{2m+r,2n+c}^{l-1,k}(x) \times w_{r,c}^{l,p,k}) - b^{l,p},$$

where K – input planes' number (as well as convolutional kernels), R and C – convolutional kernel's height and width, $w_{r,c}^{l,p,k}$ – synaptic weight with coordinates (r, c) in the convolutional kernel between k-plane of the (l-1)-layer and p-plane of the l-layer, $b^{l,p}$ – neurons' threshold of the p-plane and l-layer.

We used the sparse structure of the CNN instead of the full-connected as well as decreased the number of layers by performing the convolution and subsampling operations in each plane simultaneously [21] (Fig. 2) in order to increase the neural network's processing speed.

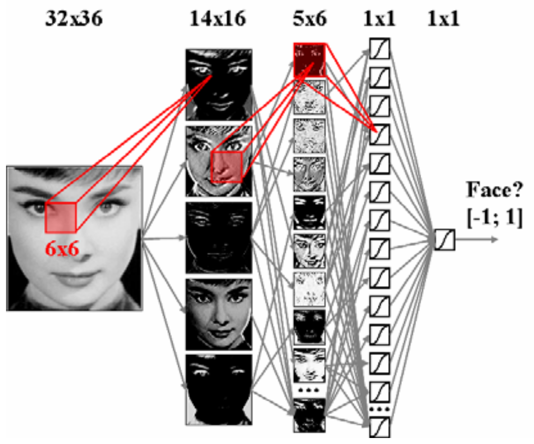


Fig. 2 – Convolutional neural network's structure for the combined cascade of classifiers.

In order to define the way of interaction between the levels of the combined cascade of classifiers we analyzed and compared two known face candidates' verification methods [5, 10-15]:

- straight verification of candidate's image by CNN,
- extended candidate's image is scanned by CNN using fixed-size window at several scale levels.

The face candidate's image is a clustering result of all multiple detections across scale and position for one candidate.

The first known verification method is simple in implementation and very fast because the number of CNN's simulations is equal to the quantity of candidates. It's used in some FD approaches with cascades of single-type classifiers: weak classifiers based on Haar-like features [10], multilayer feed-forward neural networks with local receptive fields [14], SNoW networks [15] etc. The authors' researches showed [16] that this verification method applied to the combined cascade of classifiers results in the lower detection rate (70.5% with 10 FP on CMU test set) in comparison with the FD method of P. Viola and M. Jones [10]. The detection rate increases (83.5% with 10 FP) [16] when the first known method is applied with the verification of all candidate's multiple detections. But the verification time also increases because the number of CNN's simulations for one face candidate defines as:

$$NNSim(x) = \sum_{d=0}^{D-1} WIN^{d}(x),$$

where $WIN^{d}(x)$ – d-multiple detection across scale or position for the face candidate x, D – the number of multiple detections.

The second known verification method allows to accept face candidates even in the case of inaccuracy detection of their coordinates across scale or position [3,5] and this fact is acknowledged by a high detection rate of the combined cascade of classifiers (88.8% with 9 FP on CMU test set). But authors also ascertained that the mentioned verification method is characterized by the higher computational complexity than the first known method because the number of CNN's simulations for each face candidate is

$$NNSim(x) = \sum_{s=0}^{S-1}\sum_{r=0}^{R-W}\sum_{c=0}^{C-H} WIN_{r,c}^{s}(x),$$

where $WIN_{r,c}^{s}(x)$ – a fixed-size window with the coordinates (r,c) on the face candidate's image x for s-scale level, W and H – window's width and height, R and C – width and height of a face candidate's image.

Thus two above mentioned known face candidates' verification methods have some disadvantages which don't allow their efficient applying to the combined cascade of classifiers: the first method has a low detection rate and the second one is characterized by a high computational complexity. Therefore, the authors proposed a new verification method (Fig. 3). It is based on the CNN's property to process an input image of any size at once [5].

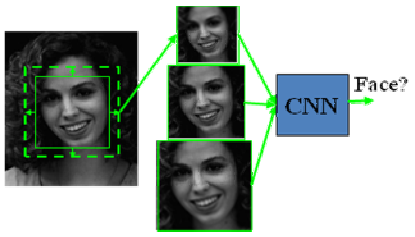


Fig. 3 – Face candidates' verification process using the proposed method.

According to the proposed face candidates' verification method the face candidate's image is extended and a pyramid of images across a scale is built for this extended area. This pyramid consists of the extended face candidate's images scaled by factor 0.8. The size of the last pyramid's image has to be not less than 32x36 (default CNN's input size). Each image from the pyramid is processed by the CNN at once instead of been scanned by a fixed-size window like in the second known verification method. After the processing of all pyramids images by a CNN the number of multiple detections is counted and a candidate is accepted as a face when the number of multiple detections is equal or greater than the threshold value which is chosen experimentally. It's obviously that the number of CNN's simulations for one face candidate is

$$NNSim(x) = \sum_{s=0}^{S-1} WIN^{s} .$$

The computational complexity's analysis of the proposed and known verification methods is showed in Table 1.

Table 1. Computational complexity comparison of the proposed and known verification methods used in the combined cascade of classifiers on CMU test set

Verification method	Total time of CNN's simulations, s	Fraction of CNN's simulations in FD process, %
First	0.7	3.1
First (with multiple detections' checking)	5.0	18.0
Second	81.6	79.3
Proposed	9.4	30.2

As we can see from Table 1, the first known verification method has the lowest computational complexity and increases the processing time of the combined cascade of classifiers by 3.1% (or 18% in the case of multiple detections' checking) on CMU test set. The largest computational complexity is inherent in the second known verification method which enlarges the processing time of the combined cascade of classifiers by 79.3%. The proposed verification method increases the processing time by 30.2%.

We also researched these verification methods according to the detection rate and number of FP (Fig. 4). The proposed verification method, as shown in Fig. 4, allows receiving in average up to 5.2% higher detection rate in comparison with the first known verification method with multiple detections' checking and only in average up to 0.2% lower detection rate as compared with the rate of the second known verification method. Taking into account the results of the abovementioned researches it is evidently that the proposed verification method is more suitable for the combined cascade of classifiers because it provides a high detection rate and allows boosting the FD process in 8.6 times in comparison with the second known verification method.

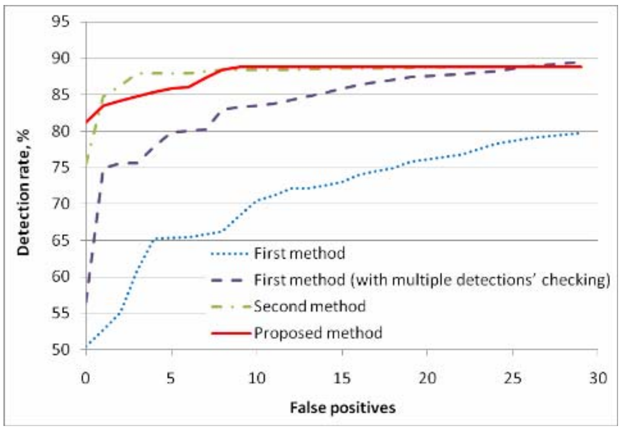


Fig. 4 – ROC-curves of the combined cascade of classifiers using different verification methods on CMU test set

The further experimental researches showed that the combined cascade of classifiers with the proposed face candidates' verification method demonstrates one of the best detection rates on CMU test set in comparison with known FD approaches (Fig. 5).

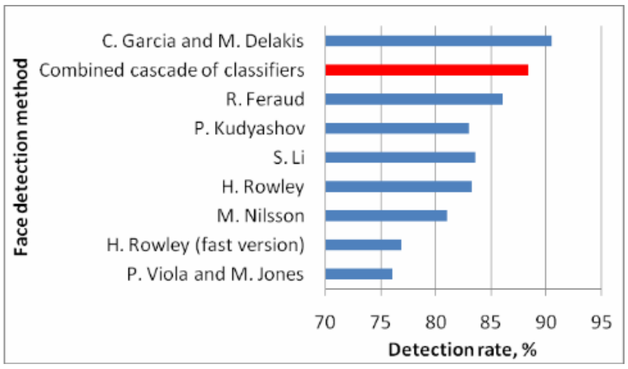


Fig. 5 – Comparison of the face detection methods' results using CMU test set at 7-10 false positives.

The improved FD method based on the combined cascade of classifiers yields only to the method of C. Garcia and M. Delakis [5] approximately on 2% by detection rate but it is 8 times faster.

3. FACE DETECTION ON COLOR IMAGES

The presence of information about skin color potentially can increase the efficiency of the FD process as it restricts the area of faces searching. As the result, it will reduce the number of false positive detections as well as the processing time of the input image [22, 23]. Thus, there is a reason to improve a method based on combined cascade of classifiers for face detection in color images. It is necessary to add an additional level of face candidates' detection to the combined cascade of classifiers before the skin color based cascade of weak classifiers [17]. Taking into account that skin color segmentation can be pixel-based or region-based, the authors make use a pixel-based segmentation. It supposes to create a classifier which will distinguish the skin color pixels from the background ones. Also it is reasonable to make use of modeling method with the explicitly defined skin regions as it is simple in implementation, fast and quite accurate [22].

There are several color spaces that can be successfully used for segmentation. For example, for RGB color space the authors use the following explicitly defined boundaries of the skin color cluster (for each of the R, G, and B channel) [24]:

% The skin color model at uniform daylight illumination

$$R > 95 \textit{ and } G > 40 \textit{ and } B > 20 \textit{ and}$$
$$\max\{R,G,B\} - \min\{R,G,B\} > 15 \textit{ and}$$
$$|R-G| > 15$$
$$\textit{and } R > G \textit{ and } R > B$$
$$OR$$

% The skin color model under flashlight lateral illumination

$$R > 220 \textit{ and } G > 210 \textit{ and } B > 170 \textit{ and}$$
$$|R-G| \le 15 \textit{ and}$$
$$R > B \textit{ and } G > B.$$

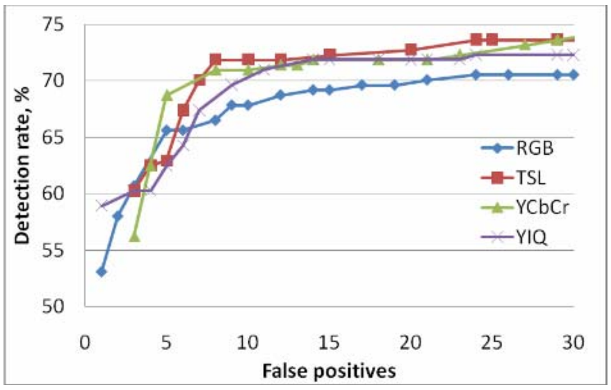
Also authors researched the TSL, YCbCr and YIQ color spaces and defined the following skin color cluster boundaries (for each of the T; Cb, Cr; I and Q channels) [23]:

$$0.45 \le T \le 0.65\ ,$$
$$85 \le C_B \le 135 \text{ and } 135 \le C_R \le 160\ ,$$
$$0.02 \le I \le 0.22 \text{ and } -0.08 \le Q \le 0.12\ .$$

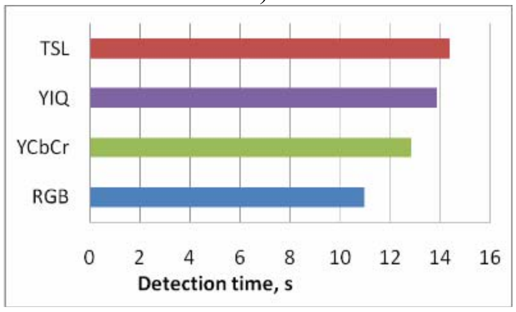
The experimental results of FD method on color images were performed on the UCD test set [25], which consists of 58 color images with 224 upright frontal faces. On the Fig. 6 there are shown the experimental results of combined cascade and skin color segmentation in RGB, TSL, YCbCr and YIQ color spaces.

As you can see from the Fig. 6a, the highest detection rate was showed by TSL-based skin color segmentation. In comparison with YCbCr, YIQ and RGB the results are better in average by 0.4%, 1.3% and 2.7% representatively. From the Fig. 6b we can conclude that the shortest time of FD is for the RGB color space.

The authors also have researched the effect on the detection rate of the combined cascade of classifiers for skin-color segmentation in two color spaces. The segmentation results are combined by logical operator AND (Fig. 7).

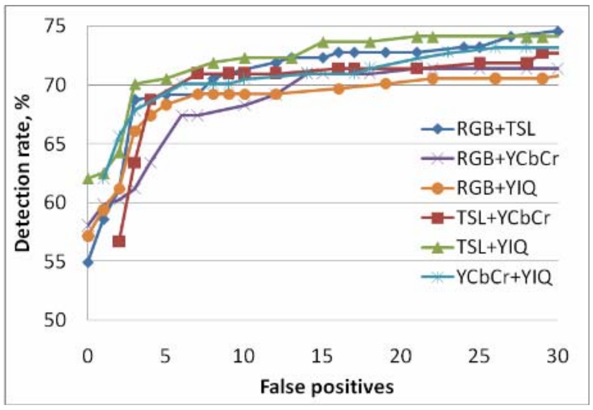


a)

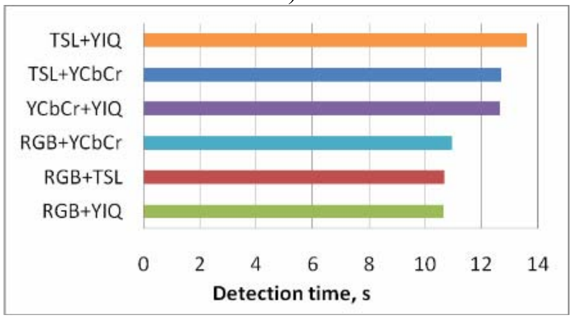


b)

Fig. 6 – ROC-curves (a) and the detection time (b) of the combined cascade of classifiers on UCD test set using different color spaces.



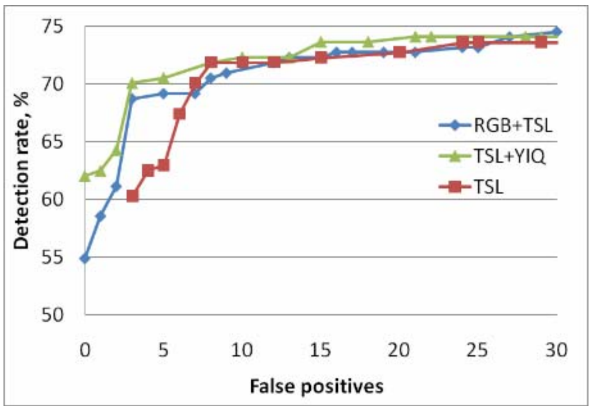
a)



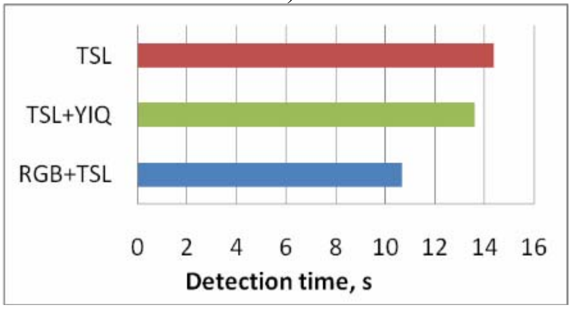
b)

Fig. 7 – ROC-curves (a) and the detection time (b) of the combined cascade of classifiers on UCD test set using combinations of color spaces.

From the FD results in Fig. 7a is it clear that to achieve high detection rate it's reasonable to use a combination of TSL+YIQ color spaces. Whereas to achieve the high speed (see Fig. 7b) we have to use a combination of RGB+TSL color spaces. It also should be noted that the RGB+TSL color spaces at the average show lower detection rate only by 1.3% in comparison with TSL+YIQ color spaces.

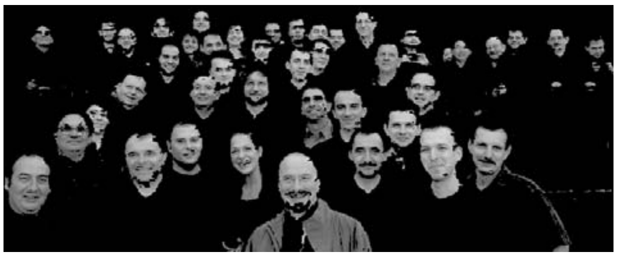


a)

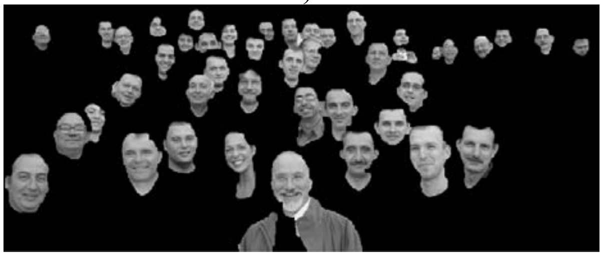


b)

Fig. 8 – ROC-curves (a) and the detection time (b) of the combined cascade of classifiers on UCD test set using one color space and their combinations.



a)

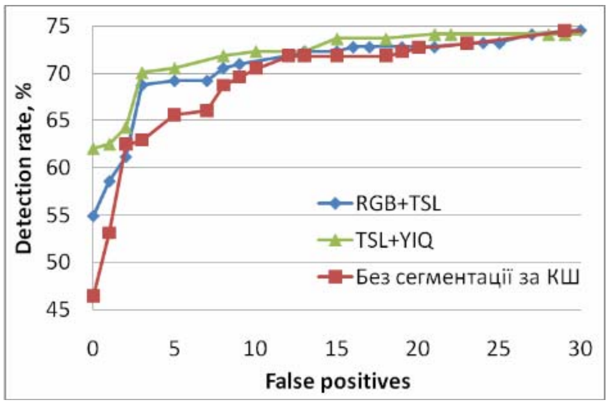


b)

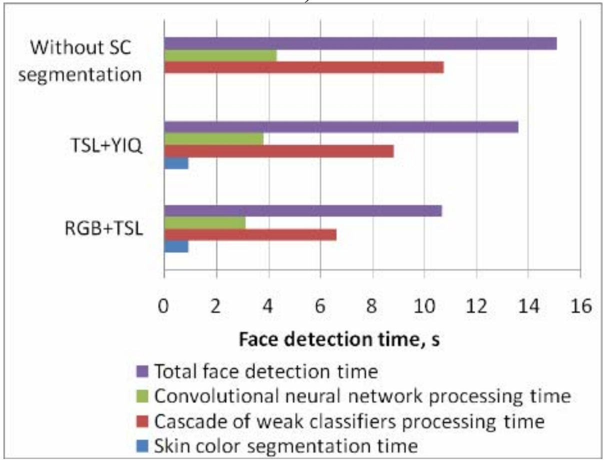
Fig. 9 – The segmented image before (a) and after (b) applying of morphological operations

The analysis of FD results with the usage of skin color segmentation in one color space and in combination of few color spaces shows (Fig. 8) that it is more reasonable to use combination of TSL+YIQ or RGB+TSL color spaces. Such approach allows obtaining higher results of FD in average by 1.7% and 0.8% accordingly. Also it allows reducing the detection time on UCD test set by 6% and 26% respectively in comparison with single TSL color space.

In order to improve the segmentation results authors used the morphological operations such as erosion, dilation and holes' filling [17, 23]. The operation of erosion will cause replacing of the boundary pixels with 0 that allows distinguishing elements from each other. The dilation will perform the backward operation by replacing boundaries pixels of background with 1. Holes' filling allows having solid segments without any background area inside. As a result of applying morphological operations the segments will be distinguished from each other, their boundaries will be rounded out and the small-size segments will be removed (Fig. 9).



a)



b)

Fig. 10 – ROC-curves (a) and the processing time (b) of the combined cascade of classifiers on UCD test set with and without skin color segmentation.

The further processing of the segmented image can be done either by passing each of the skin-like segments to the combined cascade of classifiers as the separate images or processing the entire image with the applied binary mask by combined cascade of classifiers. The mask consists of 1 where the pixel belongs to skin-color and 0 otherwise. The second approach is more preferable since it requires less computations – there are no need to find an integral image for each segment, the segments' area are not processing twice when the large segments include smaller ones. Also such approach allows detection of more face-candidates as the segments' boundaries can contain partially segmented faces and they will be processed too. In this case to avoid background processing the cascade of weak classifiers will process only those windows where the number of zero pixels is less than 70%. Such high threshold level is explained by the necessity of processing skin-color segments' boundaries since the faces are often segmented partially. The experimental results of FD method on UCD test set show that processing of the input image by combined cascade of classifiers with the applies binary mask performs by 20% faster than the processing of each segments separately.

The authors also conducted a comparative analysis of FD results by the combined cascade of classifiers with and without skin-color segmentation (Fig. 10).

As we can see from Fig.10, the improved FD method with skin-color segmentation shows higher detection rate in overage by 1.3% and 2.6%. Also it requires by 29% and 10% less time for input image processing when using RGB+TSL and TSL+YIQ color spaces respectively in comparison with FD method on grayscale images (see Section 2). Increasing of the detection rate caused by decreasing number of false positive detections since a lot of background areas where FP detections usually appear are rejected by skin-color segmentation level.

4. SELECTION OF THE FACE DETECTION SYSTEM'S PARAMETERS

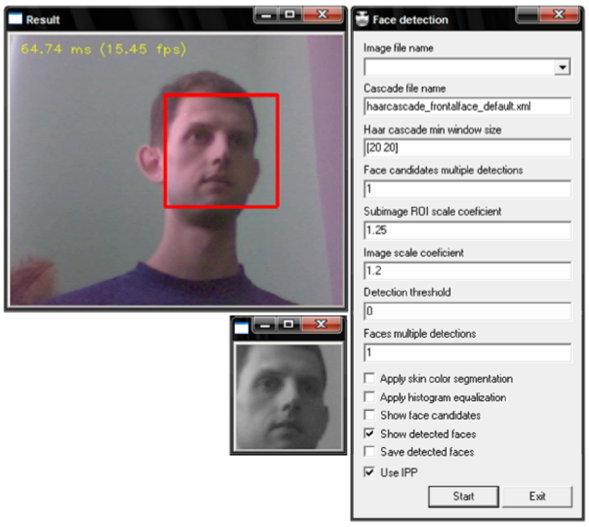
The FD system was implemented in C++ within Microsoft Visual Studio Team System 2008 environment [26] using Intel Open Computer Vision Library (OpenCV) [27] and Intel Integrated Performance Primitives (IPP) [28] (Fig. 11).

To determine the rational parameters with the maximal face detection rate, speed, and the minimal FP number authors performed a number of experiments on the test sets and in real environment, such as:

1) Determination of the number of active training epochs for CNN;
2) Comparative evaluation of the grayscale image preprocessing;
3) Determination of the multiple detections threshold values for the combined cascade of classifiers' levels at the grayscale and color image processing;
4) Test set analysis for the second kind error (the faces are not detecting);
5) Evaluation of the combined cascade of classifiers' performance while processing the video stream.

For testing there were used a Logitech QuickCam Messenger web camera, connected to the PC Intel Celeron E1200 Dual-Core 1.6GHz with 1Gb RAM.

1) In order to evaluate the required number of active training epochs for CNN for achieving acceptable detection rates and number of FP detections authors researched the dynamic of these parameters on the CMU test set during the training (Fig. 12).



64.74 ms (15.45 fps)

Image file name:

Cascade file name: haarcascade_frontalface_default.xml

Haar cascade min window size: [20 20]

Face candidates multiple detections: 1

Subimage ROI scale coeficient: 1.25

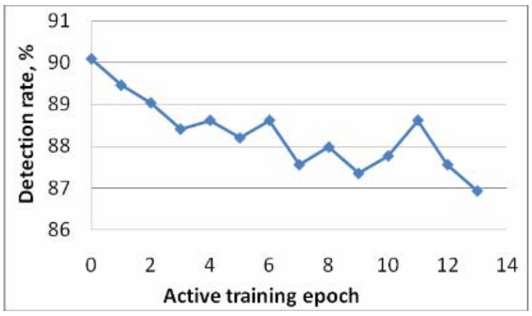
Image scale coeficient: 1.2

Detection threshold: 0

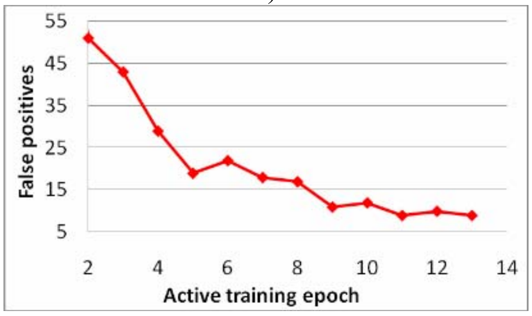
Faces multiple detections: 1

- [ ] Apply skin color segmentation
- [ ] Apply histogram equalization
- [ ] Show face candidates
- [✓] Show detected faces
- [ ] Save detected faces
- [✓] Use IPP

Fig. 11 – The main dialog box and the results of the developed face detection system



a)



b)

Fig. 12 – Dependency of the detection rate (a) and FP number (b) from CNN active training epochs on CMU test set.

As follows from the training scenario on Fig. 12, the detection rate is gradually decreasing from 90% to 87% in 14 training epochs (by 0.2% per epoch in average). Thus, based only on the detection rate dynamic's analysis, the authors can conclude that to provide highest detection rate the training time should be minimal. However, the detection rate index should be considered always in combination with the number of FP detections. That's why to achieve an acceptable number of FP detections (10-12) the CNN requires more than 8 training epochs.

2) Some FD methods [2,3] expect to receive a preprocessed image at the input of classifier. Such preprocessing includes illumination correction and pixels' histogram equalization. As there was mentioned before, there are no need in preprocessing stage for improved FD method on grayscale images. The experiments showed that the pixels' histogram equalization of the face-candidates reduces the combined cascade of classifiers' detection rate on the CMU test set in average by 6% (Fig. 13).

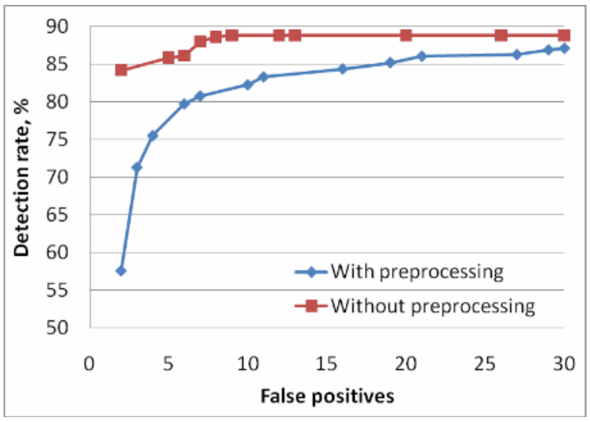


Fig. 13 – ROC-curves of the combined cascade of classifiers on CMU test set with and without input image preprocessing

3) While detecting faces no less important parameter is a threshold value of face-candidates multiple detections by cascade of weak classifiers and CNN. The threshold value affects on the correlation of detection rate and number of FP detections. It is obvious, that for the cascade of weak classifiers we should determine the value of this parameter that will provide highest detection rate. The reason is that the detection rate of cascade of weak classifiers determines the maximum possible detection rate of whole combined cascade of classifiers. The experiments showed (Fig. 14) that for cascade of weak classifiers it is reasonable to choose a threshold value of face-candidates multiple detections equal to 1. Such value allows achieving of the highest detection rate up to 91%.

The similar researches on determination of the multiple detections threshold value (*MinMD*) were also conducted for CNN (Fig. 15). As illustrated on Fig. 15, the maximum detection rate is achieved by setting the minimal number of multiple detections *MinMD*=1. In this case the detection rate is higher in comparison with other multiple detections values in average by 0.2-1.4%. We should mention also that the usage of smaller *MinMD*=1 allows improving of the combined cascade of classifiers' performance due to applying the face acceptance system which is implemented in face-candidates verification module.

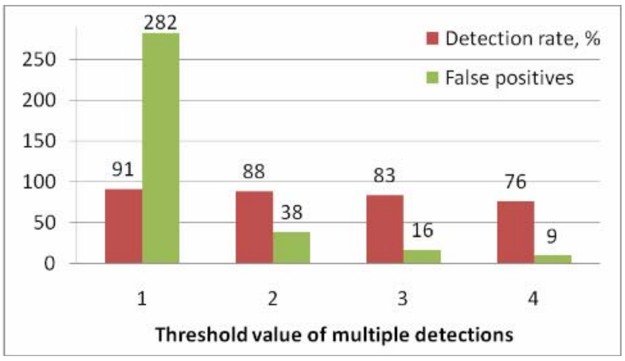


Fig. 14 – The detection rate of the cascade of weak classifiers on CMU test set with different multiple detections threshold values.

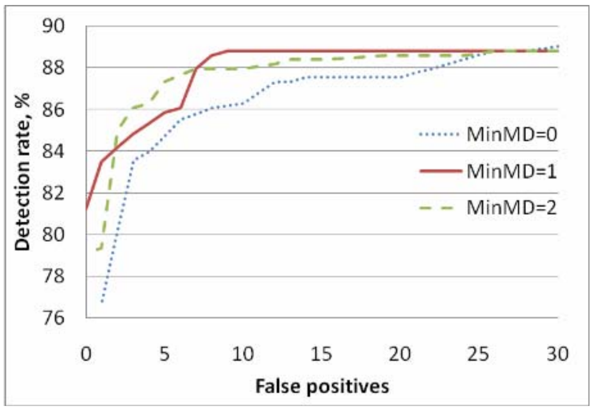


Fig. 15 – ROC-curves of the combined cascade of classifiers on CMU test set with different multiple detections threshold values for CNN.

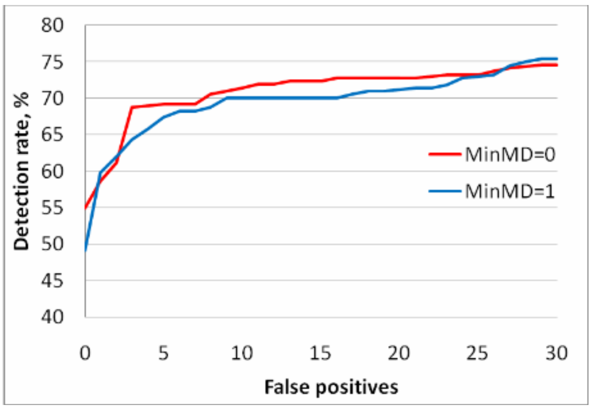


Fig. 16 – ROC-curves of the combined cascade of classifiers on UCD test set for different multiple detections threshold values for CNN.

Taking into account that preliminary experiments on determination of the minimum number of multiple detections were performed on CMU test set with the grayscale images, it is expedient to determine the threshold value of this parameter on color images also for the cases where skin-color segmentation is performed. The experimental results were performed on UCD test set (Fig. 16).

It is more reasonable to set the threshold value to 0 when processing the color images by CNN. This allows to achieve higher detection rate on average by 1.3% than at *MinMD*=1 (see Fig. 16). Reducing the minimum number of multiple detections while processing the color images (in other words the stringency of cascade of combined classifiers) is explained by additional applying of skin color segmentation which rejects much of the background – potential FP detections.

4) While analyzing the error of the second kind it is necessary to keep in mind that the combined cascade of classifiers is specified for frontal horizontal faces' detection and the possible invariance to in- and out-of-plane rotations is determined by the nature of classifiers (cascade of weak classifiers or CNN) and the train set. Particularly, the train set consist samples with rotations within [-40^0; +40^0] out of the plane, i.e. close to half profile, and the slopes in range [-20^0; +20^0]. Therefore, it is obviously that the combined cascade of classifiers should detect faces in these boundaries from the test set. Fig. 17 illustrates the examples of faces from CMU test set which were classified by mistake as non-faces.

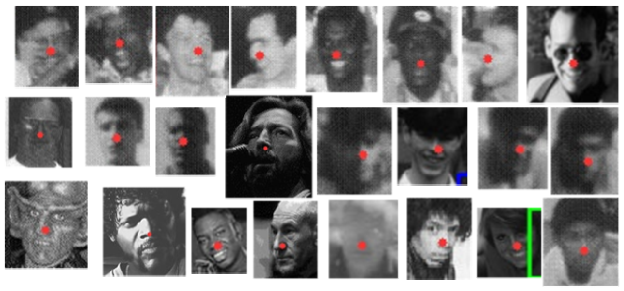


Fig. 17 – Some positive examples from CMU test set erroneously classified as non-faces by the combined cascade of classifiers.

Some erroneously classified faces have bigger in- and out-of-plane rotation angles as it was provided in train set. Also some of them are blurred or obtained in difficult illumination conditions.

5) The combined cascade of classifiers' performance analysis of video flow processing showed 15-20 fps on the 352x288 pixels input image with 20x20 pixels minimal window size.

5. CONCLUSIONS AND FUTURE RESEARCHES

Human face detection method for grayscale images is improved using the combined cascade of classifiers which consists of Haar-like features' cascade of weak classifiers and convolutional neural network. The new face candidates' verification method for the combined cascade is proposed which uses the property of the convolutional neural network to handle an input image of any size at once. The improved face detection method based on the combined cascade of classifiers yields only to the method of C. Garcia and M. Delakis [5] approximately on 2% by detection rate but it is 8 times faster.

The improved face detection method is extended also to process color images by additional using of the mixture of two color spaces for skin color segmentation. The application of TSL+YIQ or RGB+TSL color spaces' combinations allows obtaining higher face detection results in average by 1.7% and 0.8% on UCD test set. Also it allows reducing the detection time by 6% and 26% respectively in comparison w single TSL color space.

The rational parameters of the face detection system are determined during the series of experiments in order to maximize a face detection rate and speed as well as to minimize the number of false positives.

Promising future research directions include the parallelization of the combined cascade of classifiers training using the computing cluster and application of the additional face candidates' detection means like motion segmentation.

5. REFERENCES

[1] Yang M. Recent Advances in Face Detection // IEEE ICPR 2004 Tutorial. – Cambridge, United Kingdom, 2004. – 93 p.

[2] Poggio T., Sung K. Example-based learning for view-based human face detection // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 1998. – Vol. 20, No.1. – P. 39-51.

[3] Rowley H., Baluja S., Kanade T. Neural network-based face detection // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 1998. – Vol. 20. – P. 22–38.

[4] Yang M., Roth D., Ahuja N. A SNoW-Based Face Detector // Proceedings of Advances in Neural Information Processing Systems 12 (NIPS 12). – 2000. – P. 855-861.

[5] Garcia C., Delakis M. Convolution Face Finder: A Neural Architecture for Fast and Robust Face Detection // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2004. – Vol. 26, Issue 11. – P. 1408-1423.

[6] Osuna E., Freund R., Girosi F. Training Support Vector Machines: An Application to Face Detection // Proc. of IEEE Conf.

Computer Vision and Pattern Recognition. – 1997. – P. 130-136.

[7] Romdhani S., Torr P., Schlkopf B., Blake A. Computationally efficient face detection // Proceedings of ICCV. – 2001. – vol. 1. – P. 695–700.

[8] Heisele B., Serre T., Prentice S., Poggio T. Hierarchical classification and feature reduction for fast face detection with support vector machines // Pattern Recognition. – 2003. – 36(9). – P. 2007–2017.

[9] Schneiderman H., Kanade T. Probabilistic Modeling of Local Appearance and Spatial Relationships for Object Recognition // Proceedings of IEEE Conf. Computer Vision and Pattern Recognition. – 1998. – P. 45-51.

[10] Viola P., Jones M. Robust Real-Time Face Detection // International Journal of Computer Vision. 2004. – Vol. 57, No. 2. – P. 137–154.

[11] Li S., Zhang Z. FloatBoost Learning and Statistical Face Detection // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2004. – Vol. 26, No. 9. P. 1112-1123.

[12] Kudryashov P. Hybrid Human Face Detection Algorithm / P. Kudryashov, S. Fomenkov // Information Technologies. – 2007. – №10. – P. 20-23 (in Russian).

[13] Vetter T., Rätsch M., Romhani S. Efficient Face Detection by a Cascaded Support Vector Machine using Haar-like Features // Proceedings of The 26th German Association for Pattern Recognition Symposium (DAGM'04). – Tübingen (Germany), 2004. – P. 62-70.

[14] Zuo F., With P. Cascaded Face Detection Using Neural Network Ensembles // EURASIP Journal on Advances in Signal Processing. – 2008. – Vol. 2008, Issue 1. – 13 p.

[15] Nilsson M., Nordberg J., Claesson I. Face Detection using Local SMQT Features and Split up Snow Classifier // Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2007). – 2007. – Vol. 2. – P. 589-592.

[16] Paliy I. Human Face Detection Methods Using a Combined Cascade of Classifiers / I. Paliy // Computing. – 2008. – Vol. 7, Issue 1. – P. 114-125 (in Ukrainian).

[17] Paliy I. Face Detection Method and Mean for Color Images Effective Processing / I. Paliy // Artificial Intelligence. – 2008. – Vol. 4. – P. 402-411 (in Ukrainian).

[18] Freund Y., Schapire R. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting // Computational Learning Theory. – Springer-Verlag. – 1995. – P. 23–37.

[19] LeCun Y., Bottou L., Bengio Y. Gradient-Based Learning Applied to Document Recognition // Intelligent Signal Processing, IEEE Press. – 2001. – P. 306-351.

[20] Wasserman A. Neural Computing: Theory and Practice. – New York: Van Nostrand Reinhold. – 1989. – 230 p.

[21] Simard P., Steinkraus D., Platt J. Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis // Seventh International Conference on Document Analysis and Recognition (ICDAR'03). – Vol. 2. – 2003. – P. 958.

[22] Vezhnevets V., Sazonov V., Andreeva A. A Survey on Pixel-Based Skin Color Detection Techniques // Proceedings of Graphicon-2003. – Moscow (Russia), 2003. – P. 85-92.

[23] Paliy I. Improved Neural Network-based Face Detection Method using Color Images / I. Paliy, Y. Kurylyak, A. Sachenko, K. Madani, A. Chohra // Proceedings of the Third International Workshop on Artificial Neural Networks and Intelligent Information Processing (ANNIIP 2007). – Angers (France), 2007. – P. 107-114.

[24] Peer P., Kovac J., Solina F. Human Skin Colour Clustering for Face Detection // EUROCON 2003 - International Conference on Computer as a Tool. – Ljubljana (Slovenia), 2003. – Vol. 2. – P. 144-148.

[25] Sharma P., Reilly R. A Color Face Image Database for Benchmarking of Automatic Facial Detection Algorithms // Proceedings of 4th European Conference of Video/Image Processing and Multimedia Communications. – 2003. – P. 423-428.

[26] Microsoft Visual Studio Team System 2008 product page: http://msdn.microsoft.com/en-us/vsts2008/products/default.aspx.

[27] OpenCV library: http://sourceforge.net/projects/opencv/.

[28] Intel IPP library: http://www.intel.com/cd/software/products/asmo-na/eng/302910.htm.

Yuriy Kurylyak *received his Bachelor Degree in computer engineering (2005) and his Master Degree in computer systems and networks (2006) from Ternopil National Economic University (TNEU) Ternopil, Ukraine.*

From January till November 2006 he worked as an engineer at the Computer-Aided Design Laboratory, Faculty of Computer Information Technologies, TNEU. Since November

2006 he is a Ph.D. Student. From May 2007 he is a junior scientist at the Research Institute of Intelligent Computer System (ICS), TNEU. He is a member of Intelligent Robotic Systems Research Group at ICS.

His main research interests are Image and Video Processing, Motion Detection, Artificial Intelligence, Neural Networks.

***Ihor Paliy**. Scientific associate of Research Institute of Intelligent Computer Systems, Ternopil National Economic University (TNEU), Ternopil, Ukraine.*

Received the specialist's diploma of Information Systems in Management in 2002, master diploma in Economic Cybernetics in 2003from TNEU. From 2004 till now he is a PhD. student of the Department of Information Computing Systems and Control, TNEU.

His research interests are computer vision, pattern recognition, image processing, machine learning.

***Anatoly Sachenko** is Professor and Head of the Department of Information Computing Systems and Control and Director of American-Ukrainian Program in Computer Science, Ternopil State Economic University. He earned his B.Eng. Degree in Electrical Engineering at L'viv Polytechnic Institute in1968 and his PhD Degree in Electrical Engineering at L'viv Physics and Mechanics Institute in 1978 and his Doctor of Technical Sciences Degree in Electrical and Computer Engineering at Leningrad Electrotechnic Institute in 1988. Since 1991 he has been Honored Inventor of Ukraine, since 1993 he has been IEEE Senior Member.*

His main Areas of Research Interest are Implementation of Artificial Neural Network, Distributed System and Network, Parallel Computing, Intelligent Controllers for Automated and Robotics Systems. He has published over 300 papers in areas above.

***Dr. Amine Chohra.** Received his Doctorate es-sciences in 1999 from Ecole Nationale Polytech-nique, Algiers (Algeria). He has been a member of Artificial Intelligence and Robotics Laboratory (LRIA) of CDTA from 1991 to 1999. From 1999 to 2001, he worked as post doctoral researcher, with Behavior Engineering team of AiS-GMD, Sankt Augustin (Germany) and with Dependable Computing Group of IEI-CNR, Pisa (Italy), respectively. From 2001 2003, he has been teacher/researcher at Orleans University, ENSI de Bourges (France), and a member of Vision and Robotics Laboratory (LVR / UPRES EA 2078).*

Since September 2003, he is Assistant Professor at Senart Institute of Technology of PARIS XII University, Lieusaint (France) and a staff member of Image, Signal, and Intelligent Systems Laboratory (LISSI / EA 3956) of this University.

His research interests are information processing systems, knowledge based systems, hybrid intelligent systems, soft computing, machine learning and decision-making, pattern recognition and computer aided diagnosis.

***Prof. Kurosh Madani**. Received his Ph.D. degree in Electrical Engineering and Computer Sciences from University PARIS XI, Orsay, France, in 1990. From 1989 to 1990, he worked as assistant professor at Institute of Fundamental Electronics of PARIS XI University. In 1990, he joined Creteil-Senart Institute of Technology of University PARIS XII – Val de Marne, Lieusaint, France, where he worked from 1990 to 1998 as assistant professor. In 1995, he received the DHDR Doctor Habilitate degree (senior research Dr. Hab. degree) from University PARIS XII – Val de Marne. Since 1998 he works as Chair Professor in Electrical Engineering of Senart Institute of Technology of University PARIS XII.*

From 1992 to 2004 he has been head of Intelligence in Instrumentation and Systems Laboratory (I2S / JE 2353) located at Senart Institute of Technology. Since 2005, he is head of one of the three research groups of Image, Signal and Intelligent Systems Laboratory (LISSI / EA 3956) of PARIS XII University. He has worked on both digital and analog implementation of processors arrays for image processing, electro-optical random number generation, and both analog and digital ANN implementation.

His current research interests include large ANN structures modeling and implementation, hybrid neural based information processing systems and their software and hardware implementations, design and implementation of real-time neuro-control and neural based fault detection and diagnosis systems. Since 1996 he is a permanent member (elected Academician) of International Informatization Academy. In 1997, he was also elected as Academician of International Academy of Technological Cybernetics.

computing@computingonline.net
www.computingonline.net

ISSN 1727-6209
International Journal of Computing

AUTOMATIC DETECTION OF SPINAL DEFORMITY BASED ON STATISTICAL FEATURES FROM THE MOIRE TOPOGRAPHIC IMAGES

Hyoungseop Kim [1], Joo Kooi Tan [1], Seiji Ishikawa [1], Takashi Shinomiya [2]

[1] Department of Control Engineering, Kyushu Institute of Technology,
1-1, Sensui, Tobata, Kitakyushu 804-8550,
Japan, kim@cntl.kyutech.ac.jp
[2] Nikon Co. LTD., Japan

Abstract: *Spinal deformity is one of a disease mainly suffered by teenagers during their growth stage particularly from element school to middle school. There are many different causes of abnormal spinal curves, but all of them are unknown. To find the spinal deformity in early stage, orthopedists have traditionally performed on children a painless examination called a forward bending test in mass screening of school. But this test is neither objective nor reproductive, and the inspection takes much time when applied to medical examination in schools. To solve this problem, a moire method has been proposed which takes moire topographic images of human backs and checks symmetry/asymmetry of their moire patterns. In this paper, we propose a method for automatic judgment of spinal deformity which is obtained moire topographic images based on statistical features on the moire image. Statistical feature of asymmetry degrees are applied to train employing the classifier such as Artificial Neural Network, Support Vector Machine, Self-Organization Map and AdaBoost.*

Keywords: *Moire Topographic Image, Spinal Deformity, SVM, ANN, SOM, AdaBoost.*

1. INTRODUCTION

Spinal deformity is one of a serious disease, mainly suffered by teenagers. It is tends to run in families and is more common in females than males during their growth stage. There are many different causes of spinal deformity such as congenital, kyphosis (curvature of the spine with the convexity pointing toward the back), but all of them are unknown. Although the spine does curve from front to back side it should not curve lateral. A side-to-side called scoliosis and it may take the shape of an 'S' or 'C' character. The difficulty because of not accompanied by the subjective symptom such as pains the early stage detect and the early treatment becomes a problem. When one suffers from a spinal deformity, in severe case, it is associated with pain and it requires surgical treatment. The treatment of spinal deformity depends on the location and degree of curvature. Slight curves usually require no treatment, but as the curve progresses the treatment is required because the size of chest cavity diminish, it causes pain and decrease in lung.

To find the spinal deformity in early stage, orthopedists have traditionally performed a painless examination which called the forward bending test in mass screening of school. In the forward bending test, mainly medical doctor checks 5 points such as rib hump, lumbar hump, and asymmetric degree on the shoulder and west line. But this test is neither objective nor reproductive, and the inspection takes much time when applied to medical examination in school screening. To solve such various problems, a moire method [1-2] has been proposed which takes moire topographic images of human subject backs and checks symmetry/asymmetry of the moire patterns in a two-dimensional way. Fig. 1 shows an abnormal moire and its X-ray image. Asymmetrical moire patterns are appear on the subject backs. By using the moire image, the diagnosis efficiency of spinal deformity in the mass screening improved. However, the burden of the doctor who diagnoses a large amount of moire image is still remained. Then, the necessity of the image diagnosis support by using a computer is requested from the medical site. In order to achieve efficiency in diagnostic work by reducing reading time and to improve the accuracy, many systems were proposed nowadays. To detect the spinal deformity, some algorithms are proposed [3-8]. To improve the accuracy and efficiency of the primary visual screenig for the detection of spinal deformity of early stage, coputer aided dianosis (CAD) is still required.

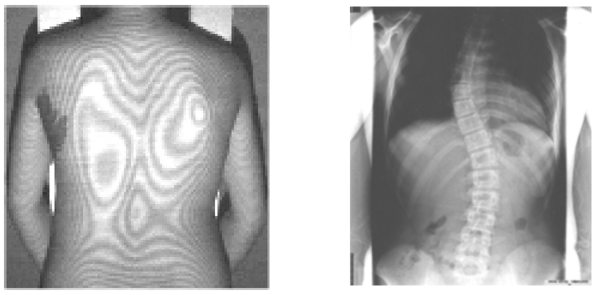


Fig. 1 – An example of moire image and its X-ray image; (a) shows an moire image, (b) shows obtained a X-ray image of (a).

In the present paper, we propose a technique for automatic detection of spinal deformity from moire topographic images by using statistical feature on the moire image. In the first step, once the original moire images is fed into computer, the middle line of the subject's back is extracted on the moire image employing the approximate symmetry analysis[9]. Regions of interest (ROIs) are automatically selected on the moire image from its upper part to the lower part and the middle line of the subject's back. Then the four asymmetry degrees are calculated from obtained ROIs. Numerical representation of the degree of asymmetry, displacement of local centroids and difference of gray value, are calculated between the right-hand side and the left-hand side regions of the moire images with respect to the extracted middle line. Feature of four asymmetry degrees (mean value and standard deviation from the each displacement) from the right-hand side and left-hand side rectangle areas apply to train the artificial neural network (ANN), support vector machine (SVM), self-organization map (SOM) and AdaBoost. In general, their algorithms can be easily introduce to classify unknown data and also can yield higher classification accuracy when the features used are not well separated. We have already proposed some techniques for automatic classifying the spinal deformity by using linear discliminant function (LDF). To increase the classification rates and avoiding the misclassification, we introduce the classifiers by using statistical features from the moire images.

2. EXTRACTION OF MIDDLE LINE

Generally, the moire stripes show symmetric patterns on the normal subject's backs. But when one becomes spinal deformity, asymmetric moire pattern appears on the moire image. In the diagnostic of imaging by using the moire method, asymmetry degree are evaluated on the moire images, so it is effective to make the asymmetry degree on the moire image in the visual screening.

To analyze the asymmetric of moire pattern, the middle line is extracted based on approximately symmetry analysis technique [9]. The approximate symmetric axis can be found by superposing the original and the reflected original image (mirror image). The best position of the superposing is determined, by evaluating the difference image which is obtained from the original and the mirror image. We adjusted to the position in which the difference of the density of a pixel values are minimized. The approximate symmetric axis is represented by the perpendicular bisector of the center of gravity of the original and the mirror image.

We assume an original moire image is $f(x,y),(x,y) \in R$, and its reflected image is represented by $f^r(x,y)$, $(x,y) \in R^r$. The $f^r(x,y)$ is superposed onto the $f(x,y)$ by parallel translation $c=(c_x,c_y)$, T is a rotation transform and rotation θ to find the best match in eq.(1). In this paper, we assume that θ=0 in eq.(2), because the moire images are captured normally straight using position-supporter so that their middle lines remain vertical.

$$D_{axis} = \min_T \sum_{(x,y)\in R\cup R^r} \left| f(x,y) - Tf^{'}(x,y) \right| \tag{1}$$

$$T = \begin{pmatrix} \cos\theta & \sin\theta & c_x \\ -\sin\theta & \cos\theta & c_y \\ 0 & 0 & 1 \end{pmatrix} \tag{2}$$

3. EXTRATION OF ASYMMETRIC FEATURES

The ROIs are selected by using pre-processing technique for extracting the asymmetrical features by the following way.

Within the region R and at a certain position $y=j$, two rectangle areas are defined, as shown in Fig.2, at symmetric locations with respect to the middle line $x=m$. The width R_x of the rectangle area is defined by,

$$R_x = \min(m-l, r-m). \tag{3}$$

Here m is the middle line which is extracted above mentioned, l and r are minimum frequency of the left- and right-hand side on the histogram, respectively. On the other hand, height of the area is defined empirically.

Let us denote the rectangle areas of the left-hand side and right-hand side at $y=i$ by A_i^l and A_i^r, respectively (See Fig.3). Here i=1,2,…,N. The centroids of A_i^l and A_i^r are denoted by $G_l(x_l,y_l)$ and

$G_r(x_r,y_r)$, respectively. The centroid $G_l(x_l,y_l)$ is reflected with respect to the middle line $x=m$ into the region A_i^r and denoted by $G_l^*(x_l^*,y_l^*)$. The distance E between $G_l^*(x_l^*,y_l^*)$ and $G_r(x_r,y_r)$ is calculated by,

$$E=\sqrt{\left(x_l^*-x_r\right)^2+\left(y_l^*-y_r\right)^2}\ . \qquad (4)$$

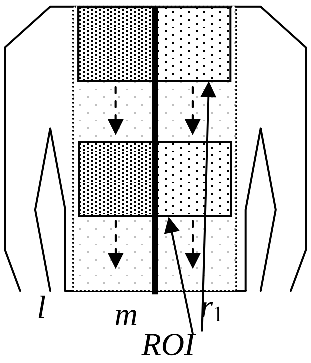


Fig. 2 – Rectangle areas in the region of interest.

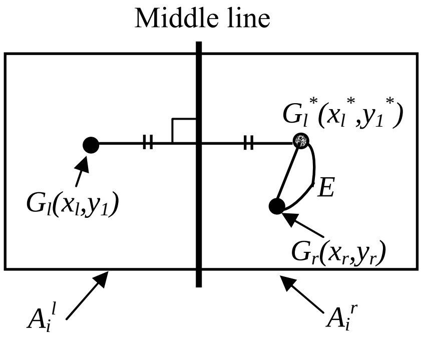


Fig. 3 – Calculation areas for local centroids.

The mean μ_E and standard deviation σ_E of the values E (i=1,2,…,N) are employed as some of the features representing the degree of asymmetry of the moire image in calculation rectangle area. The expressions are shown as follows.

$$\begin{cases}\mu_E=\dfrac{1}{N}\sum_{i=1}^{N}E\\ \sigma_E=\sqrt{\dfrac{1}{N}\sum_{i=1}^{N}\left(E-\mu_E\right)^2}\end{cases} \qquad (5)$$

Furthermore, in the same rectangle area in figure 3, the difference of gray value D on the right- and left-hand side are calculated by,

$$D=\left|r_d-l_d\right|. \qquad (6)$$

Here, r_d and l_d are shown the mean value of the gray value on the right- and left-hand side in the region in figure 3, respectively. The mean μ_D and standard deviation σ_D of the difference of gray values D (i=1,2,…,N) are employed as the other features representing the degree of asymmetry of the moire image in calculation rectangle area. The expressions are shown as follows.

$$\begin{cases}\mu_D=\dfrac{1}{N}\sum_{i=1}^{N}D\\ \sigma_D=\sqrt{\dfrac{1}{N}\sum_{i=1}^{N}\left(D-\mu_D\right)^2}\end{cases} \qquad (7)$$

4. CLASSIFICATION METHOD

The mean value and the standard deviation of the difference of the center of gravity and difference of gray value on the right-hand and left-hand side area are obtained as the statistical features. In order to detecting the unknown moire image, we have tried the ANN, SVM, SOM and AdaBoost techniques. We compared the classification performance of the four pairs.

ANN is used a useful technique for the pattern classification. This technique provided a method for the automatic spinal deformity. It is necessary for input layers, which extracted numerical feature. In order to classify the unknown moire image, the four features (mean values and standard deviation in eq.(5), (7)) which is obtained above mentioned are used for training by using the back propagation in ANN. Our ANN is consist of three layers, which include four inputs neurons, three hidden neurons and two output neurons for training by using back propagation on ANN. Finally, unknown moire images are discriminated as normal or abnormal case automatically.

The SVM [10, 11] is a supervised learning technique from the field of machine learning applicable to both classification and regression. The SVM is a set of related supervised learning methods used for classification. It is an optimization algorithm for the problem of pattern recognition. Some free software also provided methods for assessing the generalization performance efficiently. It was worked out for linear two-class classification with margin, which has the minimal distance from the separating hyper- plane to the closest data points. SVM learning machine seeks for an optimal separating hyper plane, where the margin is maximal. In this method, to classify the unknown moire images, we implement the SVM technique employing four statistical feature vectors from the left-hand side and right-hand side of rectangle areas by using the eq.(5) and eq.(7).

SOM [12] is a data visualization technique invented by T. Kohonen which reduces the dimensions of data through the use of self-organizing neural networks. In this study, we applied our method to the SOM for clustering the normal and abnormal moire image. In our method, in order to detect the unknown moire images, we implement the SOM technique employing four feature vectors from the left-hand side and right-hand side of rectangle areas (in eq.(5) and eq.(7)).

AdaBoost [13, 14] is one of the most successful and popular learning algorithms as the useful Boosting technique which is a classification algorithm designed to construct a strong classifier from a weak learning algorithm. It is a meta-algorithm which can be used in learning algorithms to improve their performance. The algorithm achieved good performance as a classifier for face detection on pattern recognition field [15]. Boosting makes a learning machine different as the weight of the exercise is changed one after another, the technique which composes the learning machine that these are combined and accuracy is high. In AdaBoost algorithm when the weight of the learning machine is updated, weight to the training sample misclassified with the learning machine increases, and weight to the training sample correctly classified decreases. Therefore, it might become difficult to see the whole image because data with a difficult distinction is emphatically learned.

5. EXPERIMENTAL RESULTS

Experiment was done employing 1200 real moire images which is 600 of abnormal and normal, respectively. The employed moire images are separated into two groups such as training and test data sets. As a training data for this study, we have selected randomly 400 (200 normal and abnormal cases, respectively) moire images. Three subsets containing normal cases are denoted by $G_i(i=1,2,3)$ and those containing abnormal and normal cases are denoted by $S_i(i=1,2,3)$. A set S_i is defined as $S_i = S_{ni} \cup S_{ai}$ $(i=1,2,3)$, (S_{ni} is normal cases and S_{ai} is abnormal cases). Then, according to the leave-out method, the set $S_j(j=1,2,3)$ is chosen as a training set and the set $S_k \cup S_l$ $(k \neq j, l \neq j, k \neq l)$ as a test set. The leave-one out is a method of applying the obtained criteria to the data group of the remainder for two data groups, doing the evaluation to which data is not biased.

The employed moire topographic image size is 256X256 pixels with 256 gray levels. Fig.4 illustrates experimental result by using only the SVM because other method could not output visually. In Fig.4, '◊' and '■' shows normal and abnormal data which is plotted feature space, respectively, which is obtained by SVM. Furthermore, a horizontal line shows the data number which consist of 400 of data set (1 to 200 of data number is normal set and 201 to 400 of data number is abnormal set). On the other hand, a vertical line shows the classification results. In our implemented SVM, '-' classified as normal and '+' classified as abnormal.

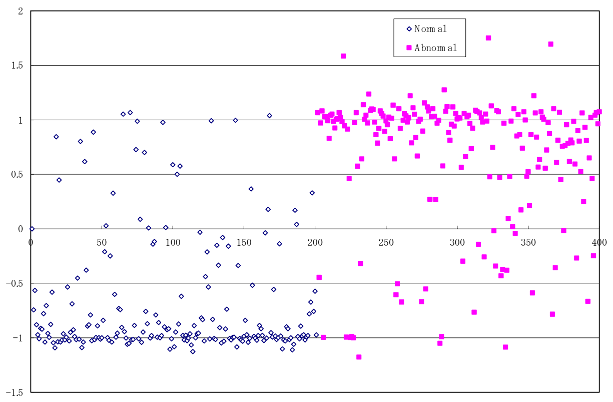
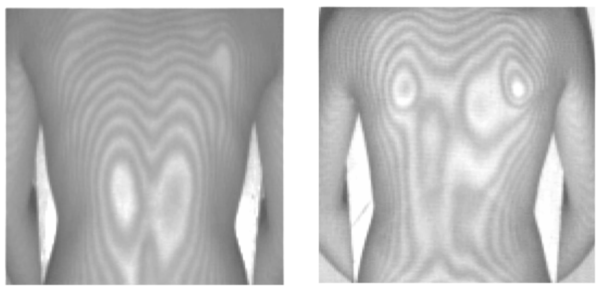


Fig. 4 – The feature space and the classified 400 data, corresponding to the G1 by using SVM in Table 1.

In Fig.5, (a) shows a normal moire image and (b) shows an abnormal moire image. Table 1 shows obtained classification rates. In the table, G_i (i=1,2,3) shows data sets, "*Normal*" shows classification rates which normal cases were classified correctly, and "*Abnormal*" shows classification rates which abnormal cases were classified correctly. Finally, "*Average*" shows the average classification rate obtained from each data group, "Ave." shows the entire average classification rate. That is, the paragraph of G1 shows the identification rate when G2 and G3 are learned as learning data, and the result of obtaining is applied to G1. As a result, on the total average, classification rate of 85.2%, 85.3%, 71.8%, and 85.6% were achieved in the ANN, SVM, SOM, and AdaBoost, respectively.



(a) normal (b) abnormal

Fig. 5 – Experimental results

5. CONCLUSIONS

In this paper, we proposed a new automatic classification method for the spinal deformity detection by using ANN, SVM, SOM, and AdaBoost method which is extracted asymmetry degree. The middle line of the subject's back is extracted on moire image employing the approximate symmetry analysis, and ROIs are automatically selected, then the asymmetry degree is calculated. Four asymmetry degrees from the right-hand and left-hand side rectangle areas which is selected as ROIs apply to train the ANN, SVM, SOM, and AdaBoost. The total average shows the classification rate of 85.2%, 85.3%, 71.8%, and 85.6% in the ANN, SVM, SOM, and AdaBoost respectively in the experiment employing 1200 moire image. In the experimental results, there is no significant difference between the performance of three classifiers such as ANN, SVM, and AdaBoost excepting SOM.

Fig.6 illustrates examples of misclassification result. In Fig.6, a normal case is classified into abnormal in (a), whereas an abnormal case is classified into normal in (b). In Fig.6, sunburn trace appears on the waist part in (a). In Fig. 6 (b), gray values subtly differ in the vicinity of an edge particularly on the shoulder part. All of the misclassified normal cases are found asymmetry of moiré patterns. This is because gray values distribution in the rectangle regions unfortunately affected symmetrically when the features were calculated. To escape from this difficulty, some other asymmetry features such as asymmetric of shoulders line or asymmetric of angle on a waist line might be taken into account in conjunction with it. These issues remain for further study.

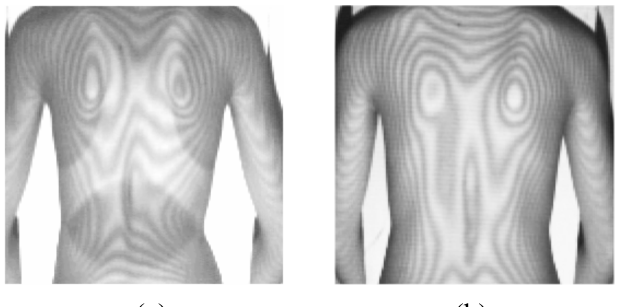

(a) (b)

Fig. 6 – Examples of misclassification: (a) Classified normal to abnormal; and (b) Classified abnormal to normal.

In this paper, we present a computer aided detection algorithm by using statistical features for detection of spinal deformity of early stage. The ANN, SVM, SOM, and AdaBoost classifier are used in the classification task with the extracted features. In the experimental results, the classification rates which abnormal cases were classified correctly are higher than the classification rates which normal cases were classified correctly. Generally, medical doctor checks the symmetric shape of right-hand and left-hand side such as waist line and shoulder line of human back. In the normal case, waist line shows almost symmetric shapes. On the other hand, in the abnormal case, asymmetric moire patterns are appeared on the waist line. To improve the classification rate in the future, we introduce a new feature such as waist line and shoulder line for the new features. That still remained as a future works.

Table 1. Classification rates [%]

	G_1	G_2	G_3	*Ave.*
ANN				
Normal	75	79.5	76	75
Abnormal	96.5	90	94	94
Average	85.8	84.8	85	85.2
SVM				
Normal	76	78.5	73	75.8
Abnormal	98	91	79.5	94.8
Average	87	84.8	84.3	85.3
SOM				
Normal	68	74.5	74	72.2
Abnormal	75	67.5	71.5	72.9
Average	71.5	71	72.8	71.8
AdaBoost				
Normal	75.5	81	80	78.8
Abnormal	95.5	90	91.5	92.3
Average	85.5	85.5	85.8	85.6

Acknowledgment

This work was supported by a Grant-In-Aid for Scientific Research on Priority Areas (18560414) from the Ministry of Education, Culture, Sports, Science and Technology, Japan. We would like to thank MD. Yoshinori Otsuka at National Sanatorium, Chiba Higashi Hospital, Japan and Hisashi Shimizu at Chiba Foundation for Health Promotion & Disease Prevention, Japan for clinical moire imae data sets and his helpful discussion and encouragement.

6. REFERENCES

[1] Y. Ohtsuka, A. Shinoto, and S. Inoue, "Mass school screening for early detection of scoliosis by use of moire topography camera and low dose X-ray imageing", *Clinical Orthopaedic Surgery*, 14, 10, pp.973-984, 1979. (in Japanese).

[2] H. Takasaki, "Moire topography from its birth to practical application", *Optics and Lasers in Engineering*, 3, pp.3-14, 1982.

[3] H. Kim, S. Ishikawa, Y. Ohtsuka, H. Shimizu, T. Sinomiya, M.A. Viergever, "Automatic scoliosis detection based on local centroids evaluation on moire topographic images of human backs", *IEEE Transaction on Medical Imag*ing, 20, 12, pp.1314-1320, 2001.

[4] M. Idesawa, T. Yatagai, T. Soma, "Scanning moiré method and automatic measurement of 3-D shapes", *Appl. Opt.*, 16, pp. 2152-2162 , 1977.

[5] H. Kim, H. Ueno, S. Ishikawa, Y. Otsuka, "Recognizing asymmetric moiré patterns for human spinal deformity detection", *Proceedings of Korea Automatic Control Conference*, pp.568-571, 1997.

[6] M. Batouche, "A knowledge based system for diagnosing spinal deformations Moire pattern analysis and interpretation", *International Conference of Pattern Recognition*, pp.591-594, 1992.

[7] I.V. Adair, M.C. Wijk, G.W.D. Armstrong, "Moiré topography in scoliosis screening", *Clin. Orthop.*, 129, p.165, 1977.

[8] H. Kim, M. Motoie, S. Ishikawa, Y. Ohtsuka, H. Shimizu, "Spinal deformity detection based on 2-D evaluation of asymmetry of moiré patterns of the human back", *Proceedings of International Technical Conference on Circuits/Systems, Computers and Communications*, pp.673-676, 1999.

[9] P. Minovic, S. Ishikawa, K. Kato, "Symmetry identification of a 3-D object represented by octree", *IEEE Trans. Patt. Anal. Machine Intell.*, PAMI-15, 5, pp.507-514, 1993.

[10] V. Vapnic, The nature of statistical learning theory, Springer-Verlag, New York, 1995.

[11] Christopher J. C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery 2, pp.121-167, 1998.

[12] T. Kohonen, Self-organizing maps, Springer-Verlag, New York, Inc., Secaucus, NJ, 1997.

[13] Y. Freund, R. E. Schapire, "Experiments with a New Boosting Algorithm", *International Conference on Machine Learning*, pp.148-156, 1996.

[14] X. Li, L. Wang, E. Sung, "A study of Adaboost with SVM weak learners", *International joint Conference on Neural Network*, pp.196-201, 2005.

[15] P. Yang, S. Shan, W. Gao, S. Z. Li, D. Zhang, "Face recognition using Ada-Boosted Gabor features", *The 6th IEEE International Conference on Automatic Face and Gesture Recognition*, pp.356-361, 2004.

Hyoungseop Kim *received his B.S., M.S. degree from the Department of Electrical Engineering at the Kyushu Institute of Technology, Japan in 1994, 1996 respectively. He received his Doc. of Eng. at the Kyushu Institute of Technology in 2001. He is currently an Associate Professor of Faculty of Engineering of the Kyushu Institute of Technology.*

His main research interests are computer vision such as tracking of moving objects for developing of security system, developing digital image analysis for medical application.

He is a member of IEEE, The Society of Instrument and Control Engineers, The Institute of Electronics, Information and Communication Engineers, and The Institute of Image Electronics Engineers of Japan.

Joo Kooi TAN *received B.E. and M.E. degrees in Computer Science and Ph.D degree in Control Engineering from Kyushu Institute of Technology. She is presently an assistant professor with faculty of Mechanical and Control Engineering in the same university.*

Her current research interests are three-dimensional shape/-motion recovery, human motion analysis, human activities recognition, and applications of computer vision.

She received the SICE Kyushu Branch Young Author's Award in 1999, the AROB10th Young Author's Award in 2004, and Young Author's Award from IPSJ of Kyushu Branch in 2004. She is a member of IEEE, The Society of Instrument and Control Engineers, and The Information Processing Society of Japan.

Seiji Ishikawa *obtained B.E., M.E., and D.E. from The University of Tokyo, where he majored in Mathematical Engineering and Instrumentation Physics. He joined Kyushu Institute of Technology and he is currently Professor of Department of Control & Mechanical Engineering. Professor Ishikawa was a visiting research fellow at Sheffield University, U.K., from 1983 to 1984, and a visiting professor at Utrecht University, The Netherlands, in 1996.*

His research interests include three-dimensional shape/motion recovery, human motion analysis, medical image analysis, and a multiple robots system. He is a member of IEEE, The Society of Instrument and Control Engineers, The Institute of

Electronics, Information and Communication Engineers, and The Institute of Image Electronics Engineers of Japan.

***Shinomiya Takashi** received his PhD, from Graduate School of Kyushu Institute of Design (Kyushu University). He is currently General Manager of Planning Dept., Core Technology Center, Nikon Corporation., and Part-time Professor of Ergonomics at Organization of Institute of Industrial Ecological Sciences, University of Occupational and Environmental Health, Japan. He is a member of International Ergonomics Association (IEA)*

CiSJ

computing@computingonline.net
www.computingonline.net

ISSN 1727-6209
International Scientific
Journal of Computing

A SURVEY ON WAVELET NETWORK, MULTI LIBRARY WAVELET NETWORK TRAINING, 1D-2D FUNCTION APPROXIMATION AND A NEW IMAGE COMPRESSION METHOD

Wajdi Bellil [1), Chokri Ben Amar [2), Adel M.Alimi [3)

[1) Faculty of sciences, University of Gafsa, City Zarroug, Gafsa, Tunisia wajdi.bellil@ieee.org
[2) Department of Electrical Engineering, University of Sfax, Tunisia Chokri.benamar@ieee.org
[3) Department of Electrical Engineering, University of Sfax, Tunisia Adel.Alimi@ieee.org

Abstract: *This paper presents an original architecture of Wavelet Neural Network (WNN) based on multi Wavelets activation function and uses a selection method to determine a set of best wavelets whose centers and dilation parameters are used as initial values for subsequent training library WNN for color image compression and coding which consists to transform an RGB image into Luminance-Chrominance space and then segment the luminance in a set of m blocks n by n pixels. These blocks should be transferred row by row (1D input vector) to the input of our wavelet network. Every input vector will be considered as unknown functional mapping and then it will be approximated by the network.*

Keywords: *Wavelet Neural Network, Multi Library Wavelet Neural Network, Image compression and coding, Beta wavelets.*

1. INTRODUCTION

Wavelet Neural Networks (WNN) were introduced by Zhang and Benveniste [1-3] in 1992 as a combination of artificial neural networks and wavelet decomposition. WNN have recently attracted great interest, because of their advantages over radial basis function networks (RBFN) as they are universal approximators but achieve faster convergence and are capable of dealing with the so-called "curse of dimensionality." In addition, WNN are generalized RBFN. However, the generalization performance of WNN trained by least-squares approach deteriorates when outliers are present.

Feed forward neural networks such as multilayer perceptrons (MLP) and radial basis function networks (RBFN) have been widely used as an alternative approach to functions approximation since they provide a generic black-box functional representation and have been shown to be capable of approximating any continuous function defined on a compact set in Rn with arbitrary accuracy [4]. Following the concept of locally supported basis functions such as RBFN, a class of wavelet neural networks (WNN) which originate from wavelet decomposition in signal processing has become more popular lately [5, 6, 7, 8, 9]. In addition to the salient feature of approximating any non-linear function, WNN outperforms MLP and RBFN due to its capability in dealing with the so-called "curse of dimensionality" and non-stationary signals and in faster convergence speed [10]. It has also been shown that RBFN is a special case of WNN.

This paper comprises four sections. Section 2 discusses the architecture of Multi Library Wavelet Neural Networks (MLWNN) and a new training algorithm based on selection. Section 3 contributes to Beta MLWNN and its performance function approximation. Section 4 presents a direct solution method based on wavelet networks for lossless color image compression. Finally, Section 5 gives conclusions and summary for present research work and other possibilities of future research directions.

2. THEORETICAL BACKGROUND

2.1 CLASSICAL WAVELET NEURAL NETWORK ARCHITECTURE

Wavelets occur in family of functions and each is defined by dilation ai which controls the scaling parameter and translation ti which controls the position of a single function, named the mother wavelet $\psi(x)$. Mapping functions to a time-frequency phase space, WNN can reflect the time-frequency properties of function. Given an n-element training set, the overall response of a WNN is:

$$\hat{y}(w) = w_0 + \sum_{i=1}^{N_p} w_i \Psi_i \text{ , where } \Psi_i = \Psi\left(\frac{x - t_i}{a_i}\right) \quad (1)$$

where Np is the number of wavelet nodes in the hidden layer and wi is the synaptic weight of WNN.

This can also be considered as the decomposition of a function in a weighted sum of wavelets, where each weight w_j is proportional to the wavelet coefficient scaled and shifted by ai and ti. This establishes the idea for wavelet networks [11, 12].

This network can be considered composed of three layers: a layer with Ni inputs, a hidden layer with N_p wavelets and an output linear neuron receiving the weighted outputs of wavelets. Both input and output layers are fully connected to the hidden layer.

2.2 MULTI LIBRARY WAVELET NEURAL NETWORK ARCHITECTURE

A MLWNN can be regarded as a function approximator which estimates an unknown functional mapping:

$$y = f(x) + \varepsilon \quad (2)$$

where *f* is the regression function and the error term ε is a zero-mean random variable of disturbance. Constructing a *MLWNN* involves two stages: First, we should construct a wavelet library *W={W₁, W₂,…,Wₙ}* of discretely dilated and translated versions of some mothers wavelets function $\Psi_1, \Psi_2, \ldots, \Psi_n$:

$$W_j = \left\{ \begin{array}{l} \Psi_i^j : \Psi_i^j(x) = \alpha_i \Psi^j\left(a_i(x - t_i)\right), \\ \alpha_i = \left(\sum_{k=1}^{N} \left[\Psi^j(a_i(x_k - t_i))\right]^2 \right)^{-\frac{1}{2}} \\ i = 1,\ldots, L \text{ and } j = 1,\ldots, n \end{array} \right\} \quad (3)$$

where x_k is the sampled input and L is the number of wavelets in each sub library W_j. Then select the best *M* wavelets based on the training data from multi wavelet library *W*, in order to build the regression.

$$\hat{y}(x) = \sum_{i\in I} w_i \Psi_i^1(x) + \sum_{i\in I} w_i \Psi_i^2(x) + \ldots + \sum_{i\in I} w_i \Psi_i^n(x) \quad (4)$$

2.3 AN INITIALIZATION PROCEDURE USINGA SELECTION METHOD

It is very inadvisable to initialize the dilations and translations randomly, as is usually the case for the weights of a standard neural network with sigmoid activation function. In the case of wavelet neural network and due to the fact that wavelets are rapidly vanishing functions, a wavelet may be too local if its dilation parameter is too small (it may sit out of the domain of interest), if the translation parameter is not chosen appropriately.

We propose to make use of multi library wavelet using a selection method to initialize the translation and dilation parameters of wavelet networks trained using gradient-based techniques. The procedure comprises four steps:

2.3.1 INITIALIZATION

Let Y the signal to be approximated, we have the same library that previously. This library contains N_{Mw} wavelet. We associate to every wavelet a vector whose components are the values of this wavelet according to the examples of the training sequence. We constitute a matrix V_w thus constitutes blocks of the vectors representing the wavelet of every mother wavelet:

$$V_w = \left| \begin{array}{ccccccccccccc} V_1^1(x_1) & \ldots & V_N^1(x_1) & \ldots & V_1^J(x_1) & \ldots & V_N^J(x_1) & \cdots & V_1^M(x_1) & \cdots & V_N^M(x_1) \\ V_1^1(x_2) & \cdots & V_N^1(x_2) & \ldots & V_1^J(x_2) & \cdots & V_N^J(x_2) & \cdots & V_1^M(x_2) & \cdots & V_N^M(x_2) \\ \vdots & \ldots & \vdots & \ldots & \vdots & \ldots & \vdots & \ldots & \vdots & \ldots & \vdots \\ \vdots & \ldots & \vdots & \ldots & \vdots & \ldots & \vdots & \ldots & \vdots & \ldots & \vdots \\ \vdots & \ldots & \vdots & \ldots & \vdots & \ldots & \vdots & \ldots & \vdots & \ldots & \vdots \\ V_1^1(x_N) & \ldots & V_N^1(x_{N_i}) & \cdots & V_1^J(x_N) & \ldots & V_N^J(x_{N_i}) & \cdots & V_1^M(x_{N_i}) & \cdots & V_N^M(x_{N_i}) \end{array} \right| \quad (5)$$

$$V_w(t,d) = \{V_i^j\} i=[1..N], j=[1..M] \quad (6)$$

We note by :
g(X) the constructed network,
N_w=1, the wavelet number
T={tᵢ } i=[1..N] , the translation vector
D={dᵢ } i=[1..N], the dilation vector

2.3.2 SELECTION

The library being constructed, a method of selection is applied in order to determine the most meaningful wavelet for approximation the considered signal. In general the wavelets in W are not all meaningful to estimate the signal. Let's suppose that we want to construct a network *g(x)* of wavelets with m wavelets, the problem is to select m wavelets of W.

To the first iteration, the signal is $y = Y_1$, and the vector regressor is the *Vw(t,ds)* definite by (6), the selected regressor is the one for which the absolute value of the cosine with the *Y1* signal is maximal. We define ipert1 as:

$$\text{ipert1(i,j)} = \arg \max_{i,j} \frac{\left\langle Y \middle| V_i^j \right\rangle}{\|Y\| . \|V_i^j\|}, \quad (7)$$

with i=[1..N], j=[1..M]

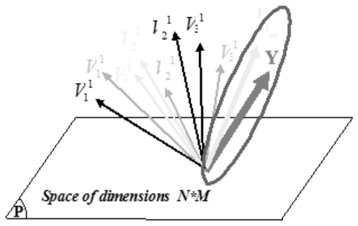


Fig. 1 – Selection of the pertinent vector.

The regressor $V_{i_{pert}}1$ can be considered like an adjustable and temporal function used to approximate Y (see figure 1). We calculate the weight w_1 defined by:

$$w_1 = \frac{Y}{V_{i_{pert}}1} \tag{8}$$

We calculate thereafter, the mean square error of training (MSET) definite by:

$$\text{MSET}(\omega_{i\,pert}, t_{i\,pert}, d_{i\,pert}) = \frac{1}{N}\sum_{k=1}^{N}(Y(k) - \omega_{i\,pert} * V_{i_{pert}})^2 \tag{9}$$

Y(k) is the desired output correspondent to the example *k*, and $\omega_{i\,pert} * V_{i_{pert}}$ is the wavelet network output to the example *k*.

2.3.3 REGRESSOR OPTIMIZATION

To optimize the regressor we used the pressure gradient method:

We notes by:

$e(x) = Y_d(x) - Y(x)$, with Yd: desired output and *Y*: the real network output.

$$\frac{\partial MSET}{\partial w_{ipert}1} = \sum_{i=1}^{Nw} e(x)\Psi\left(\frac{x - t_{ipert}1}{d_{ipert}1}\right) \tag{10}$$

$$\frac{\partial MSET}{\partial d_{iper}1} = \sum_{i=1}^{Nw} e(x) w_{_{ipert}} 1 \frac{\partial \Psi\left(\frac{x - t_{ipert}1}{d_{ipert}1}\right)}{\partial d_{ipert}1} \tag{11}$$

$$\frac{\partial MSET}{\partial t_{iper}1} = \sum_{i=1}^{Nw} e(x) w_{_{ipert}} 1 \frac{\partial \Psi\left(\frac{x - t_{ipert}1}{d_{ipert}1}\right)}{\partial t_{ipert}1} \tag{12}$$

This optimization has the advantage to be fast because we only optimize the three structural parameters of the network.

To the exit of this optimization, the parameters: $t_{i_{pert}}^{opt}1, d_{i_{pert}}^{opt}1,\ \omega_{i_{pert}}^{opt}1$ of the regressor $V_{i_{pert}}1$ are adjusted, and are solutions of the optimization problem defined by:

$$V_{i_{pert}}^{opt} = V_{i_{pert}}(\omega_{i_{pert}}^{opt}, t_{i_{pert}}^{opt}, d_{i_{pert}}^{opt}) \tag{13}$$

Considering the optimal regressor, we reset the network with this regressor that is going to replace the old in the library. The orthogonalization will be done in relation to the optimal regressor as shown in figure 2.

$$V_W = \left| \begin{array}{ccccc} V_1^1(x_1) \ \ldots & V_N^1(x_1) & \ldots V_{i_{pert}}^{opt}(x_1) \cdots & V_1^M(x_1) & \ldots \ V_N^M(x_1) \\ V_1^1(x_2) \ \cdots & V_N^1(x_2) & \cdots V_{i_{pert}}^{opt}(x_2) \cdots & V_1^M(x_2) & \cdots \ V_N^M(x_2) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ V_1^1(x_N) & V_N^1(x_{N_i}) & \cdots V_{i_{pert}}^{opt}(x_{N_i}) \ldots & V_1^M(x_{N_i}) & \cdots \ V_N^M(x_{N_i}) \end{array} \right| \tag{14}$$

After an iteration we will have:

$$\text{V}_{\text{w1}}(\text{t,d}) = \{\text{V}_\text{i}^\text{j}\}_{\text{i=[1..N], j=[1..M]}} \cup \left\{V_{i_{pert}}^{opt}\right\} \tag{15}$$

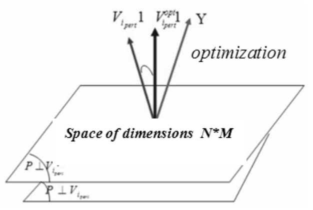


Fig. 2 – Regressor Optimization.

2.3.4 ORTHOGONALIZATION

The vectors V_i^j are always linearly independent, (because *N >> M*$_W$) and non orthogonal. The vectors V_i^j generate one sub-vector-space of *M*N* dimension. We orthogonalize the *N*M*$_S$*-1* remaining regressors, and the vector *Y*$_1$ in relation to the adjusted regressor $V_{i_{pert}}^{opt}1$:

$$V_i^{j\perp} = V_i^j - \left\langle V_i^j \middle| V_{i_{pert}}^{opt}1 \right\rangle V_{i_{pert}}^{opt}1 \tag{16}$$

$$Y^{\perp}1 = Y - \left\langle Y \middle| V_{i_{pert}}^{opt}1 \right\rangle V_{i_{pert}}^{opt}1 \tag{17}$$

We make then, the updating of the library:

$$V_W = \left| \begin{array}{cccccc} V_1^{1\perp}(x_1) \ \ldots & V_N^{1\perp}(x_1) & \ldots V_1^{J\perp}(x_1) \ \ldots & V_N^{J\perp}(x_1) \ \ldots & V_1^{M\perp}(x_1) \ \ldots & V_N^{M\perp}(x_1) \\ V_1^{1\perp}(x_2) \ \cdots & V_N^{1\perp}(x_2) & \cdots V_1^{J\perp}(x_2) \ \cdots & V_N^{J\perp}(x_2) \ \cdots & V_1^{M\perp}(x_2) \ \cdots & V_N^{M\perp}(x_2) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ V_1^{1\perp}(x_N) & V_N^{1\perp}(x_{N_i}) & \cdots V_1^{J\perp}(x_N) & V_N^{J\perp}(x_{N_i}) \ \cdots & V_1^{M\perp}(x_{N_i}) \ \cdots & V_N^{M\perp}(x_{N_i}) \end{array} \right| \tag{22}$$

We obtain:

$$V_w(t,d) = \{V_i^{j\perp}\}_{(i=[1..N],\, j=[1..M]) \setminus \{ i_{pert} \}} \quad (18)$$

$Y^{\perp}1$ and $\{V_i^{j\perp}\}$ are respectively what remains the signal and regressors in the space orthogonal to $V_{i_{pert}}^{opt}1$.

The model being to this stage, $g(X) = \omega_i 1 * V_{i_{pert}}^{opt} 1$, is shown in figure 3.

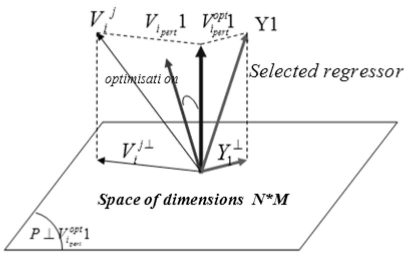


Fig. 3 – Orthogonal projection on the optimal regressor.

To the following iteration we increment the number of wavelet *N*$N_w=N_w+1$. We applied the same stages decry above. Let's suppose achieved *i-1* iterations: we did *i - 1* selections, optimizations, and orthogonalizations. To get the *i-1* adjusted regressors we reset i-1 parameter of the network.

At the end of the iteration *i -1*, the expression of the network *g(X)*, is given by:

$$g(X) = \sum_{i=1}^{i-1} \omega_i^{opt} * V_{i_{pert}}^{opt} \quad (19)$$

We have Nw-i+1 régressors to represent the signal Yi in a space of N*M - i +1 dimension orthogonal to ($V_{i_{pert}}^{opt}1$,..., $V_{i_{pert}}^{opt}i-1$)

We apply the same procedure of selection as previously, the indication iperti of the selected regressor is the one for which the absolute value of the cosine with the signal Yi is maximal; iperti is given by (7)

Finally and after N iteration, we construct a wavelet network with N wavelet in the hidden layer that approximates the input signal Y.

So the parameters of the network are:

$$T^{opt} = \left\{ t_{i_{pert}}^{opt} \right\}_{i_{pert}=[1..Nopt]} \quad (20)$$

$$d^{opt} = \left\{ d_{i_{pert}}^{opt} \right\}_{i_{pert}=[1..Nopt]} \quad (21)$$

$$\omega^{opt} = \left\{ \omega_{i_{pert}}^{opt} \right\}_{i_{pert}=[1..Nopt]} \quad (22)$$

The obtained model *g(X)* can be written under the shape:

$$g(X) = \sum_{i=1}^{N_{opt}} \omega_i^{opt} * V_i^{opt} \quad (23)$$

3. BETA MULTI LIBRARY WAVELET NETWORK

3.1 BETA WAVELET FAMILY

The Beta function [14] is defined as:
if p>0, q>0, (p, q) ∈ IN

$$\beta(x) = \begin{cases} \left(\frac{x-x_0}{x_c-x_0}\right)^p \left(\frac{x_1-x}{x_1-x_c}\right)^q \ if\ x \in \left[x_0, x_1\right] \\ 0 \quad else \end{cases} \quad (24)$$

Where, $x_c = \dfrac{px_1+qx_0}{p+q}$

We prove in [15] that all the derivatives of Beta function $\in L^2(\Re)$, are of class C^{∞} and satisfy the admissibility wavelet condition for *p=q*.

3.2 EXAMPLE 1:1-D FUNCTION APPROXIMATION

The first example is the approximation of a function of a single variable function, without noise, given by:

$$f(x) = \begin{cases} -2.186x - 12.864 \quad for\ x \in [-10, -2[\\ 4.246x \quad for\ x \in [-2, -0[\\ 10\exp(-0.05x - 0.5)\sin(x(0.03x + 0.7)) \quad for\ x \in [0, 10[\end{cases} \quad (25)$$

First, simulations on the 1-D function approximation are conducted to validate and compare the proposed MLWNN with the classical WNN. The input x is constructed by the uniform distribution on [-10 10]. The training sequence is composed of 101 points. The performance of the model is estimated using a test set of 101 equally spaced examples different from the training set.

We define the *NMSE* (Normalized Mean Square Error) as evaluation criteria.

$$NMSE = \frac{1}{N} \sum_{k=1}^{N} \left(\hat{f}(x_k) - y_k \right)^2 \quad (26)$$

In the following, we present the results obtained with a network of 12 Beta wavelets, chosen as mother wavelets (second and third derivative of Beta function), for training network. Figure 4 shows the initial error histogram (a) obtained when the 101 input patterns are initialized with the classical architecture and the final error histogram (b) obtained when the 101 input patterns are training after 1000 iterations. Figure 4 (c) shows the initial error histogram obtained when the 101 trainings are initialized with the initialization by selection procedure using MLWNN and the final error histogram (d) obtained when the 101 input patterns are training after 1000 iterations. We can see clearly that the initialization by selection using MLWNN leads to:

- The best result in term of NMSE,
- Less scattered results both on the training set and on the test set.
- Using multi wavelet mothers as activation function gives best approximation.

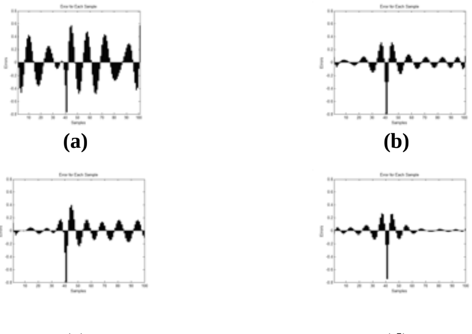


Fig. 4 – Evolution of the initial and final error for each sample after initialization using classical WNN architecture and MLWNN architecture.

Figure 5 shows the evolution of the NMSE according to the iteration; (a) shows the initial error for each sample after initialization using classical WNN architecture, (b) gives the final error for each sample after initialization using classical WNN architecture, when figure (c) and (d) show respectively the initial and final error for each sample after initialization using MLWNN architecture. We can see the superiority of the proposed initialization selection algorithm based on multi wavelet library over the classical WNN based on one mother wavelet.

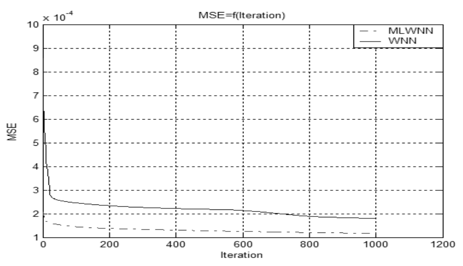


Fig. 5 – Evolution of the NMSE according to the iteration.

3.3 EXAMPLE 2: 2-D FUNCTION APPROXIMATION

The process to be modeled is simulated by a function of two variables without noise. The expression of this function is given by:

$$f(x_1, x_2) = \left(\frac{x_1+2}{2}\right)^5 \left(\frac{x_2+2}{2}\right)^5 \left(\frac{2-x_1}{2}\right)^5 \left(\frac{2-x_2}{2}\right)^5 \qquad (27)$$

In the following, we present the results obtained with a network of 9 Beta wavelets, chosen as mother wavelets (second and third derivative of Beta function), for training network. The training set contains 11x11 uniform spaced points. The test set V is constructed by 21x21 stochastic points on [-1,1]x[-1,1]. Figure 6 shows the final error histogram (a) obtained when the 121 trainings are initialized with the classical architecture initialization and the final error histogram (b) obtained when the 121 trainings are initialized with a selection procedure using MLWNN.

Fig. 6 – Final error for each sample after initialization using classical WNN architecture (a) and MLWNN architecture (b).

These results show that the effect of the classical WNN initialization is much smaller than when the wavelet centers and dilations are initialized by selection using a multi library WNN, used together with Beta wavelets, it makes wavelet neural network training very efficient because of the adjustable parameters of Beta function.

4. BETA WAVELET NEURAL NETWORK FOR LOSSLESS COLOR IMAGE COMPRESSION

The idea consists to transform an RGB image into luminance-chrominance space. Compression is achieved by determining the value of N_p. The input (Luminance) is split up into blocks or vectors of 8x8, 16x16 or 32x32 pixels. When the input vector is referred to as 1-dimensional which is equal to the number of pixels included in each block, each wavelet at the hidden layer can be represented by three parameters: weight, translation and dilation, which can also be described by three matrix of order 3x N_p. Image compression is achieved by training the network in such a way the network output scale the input and produces the optimum output value which makes the quadratic error between input and output minimum.

In accordance with the neural network structure shown in Fig.1 the encoding phase can be described as follows:

$$e(x) = y(x) - \hat{y}(x) \qquad (28)$$

The energy function is define as

$$E = \frac{1}{2}\sum_{x=1}^{X} e^2(x) \tag{29}$$

The mean square error is optimised according to these equations

$$\frac{\partial E}{\partial w_k} = -\sum_{x=1}^{X} e(x)\Psi(\tau) \tag{30}$$

$$\frac{\partial E}{\partial x_k} = -\sum_{x=1}^{X} e(x) w_k \frac{\partial \Psi(\tau)}{\partial x_k} \tag{31}$$

$$\frac{\partial E}{\partial d_k} = -\sum_{x=1}^{X} e(x) w_k \tau \frac{\partial \Psi(\tau)}{\partial x_k} = \tau \frac{\partial E}{\partial x_k} \tag{32}$$

$$with\, \tau = \frac{x - x_k}{d_k} \tag{33}$$

$$\Delta w = -\frac{\partial E}{\partial w}, \Delta t = -\frac{\partial E}{\partial t}, \Delta d = -\frac{\partial E}{\partial d} \tag{34}$$

$$\underline{w}(n+1) = \underline{w}(n) + \mu_w \Delta \underline{w} \tag{35}$$

$$\underline{t}(n+1) = \underline{t}(n) + \mu_t \Delta \underline{t} \tag{36}$$

$$\underline{d}(n+1) = \underline{d}(n) + \mu_d \Delta \underline{d} \tag{37}$$

μ is the training constant.
where $x \in$ [0, 1] denotes the normalized pixel values for RGB images with intensity levels [0, 255]. The reason for using normalized pixel values is due to the fact that wavelet networks can operate more efficiently when both their inputs and outputs are limited to a range of [0, 1] [5].

In the first phase, a set of image samples is designed to train the network via the descent gradient learning rule which uses each input vector as the desired output. This is equivalent to compressing the input into the narrow channel represented by the hidden layer and then reconstructing the input from the hidden to the output layer.

The second phase simply involves the entropy coding of the state vector Ψ at the hidden layer. Since the hidden wavelet output is real valued, quantization is required for fixed length entropy coding which is normally designed as 32 level uniform quantization corresponding to 5 bits entropy coding [9,14].

4.1 PERFORMANCE ASSESSMENTS

Around the Wavelet Neural Network, we have described the scheme to achieve image data compression can normally be assessed by considering two measurements. One is the compression ratio or bit rate which is used to measure the compression performance, and the other is mainly used to measure the quality of reconstructed images with regards to a specific compression ratio or bit rate. The definition of this measurement, however, is a little ambiguous at present [16]. In practice, there exists two acceptable measurements for the quality of reconstructed images which are *PSNR* (peak-signal-to noise ratio) and *NMSE*. Their definitions can be given, respectively, as follows:

$$PSNR = 10\log \frac{255^2}{\frac{1}{n\,N}\sum_{i=1}^{n}\sum_{j=1}^{N}\left(\overline{P_{ij}} - P_{ij}\right)^2}\left(dB\right) \tag{38}$$

$$NMSE = \frac{\sum_{i=1}^{n}\sum_{j=1}^{N}\left(\overline{P_{ij}} - P_{ij}\right)^2}{\sum_{i=1}^{n}\sum_{j=1}^{N} P_{ij}^2} \tag{39}$$

Where $\overline{P_{ij}}$ is the intensity value of pixels in the reconstructed images; and P_{ij} the intensity value of pixels in the original images which are split up into n input vectors: $x_i=\{P_{i1}, P_{i2},\ldots, P_{iN}\}$.

4.2 PERFORMANCES ACCORDING TO THE PSNR, AND NMSE FOR COLOR IMAGE COMPRESSION USING MLWNN

We use 3 RGB test images: Nature1, Nature2 and Lena. For each image we compute The PSNR, SNR and NMSE using MLWNN composed of 4 and 10 neurons in hidden layer.

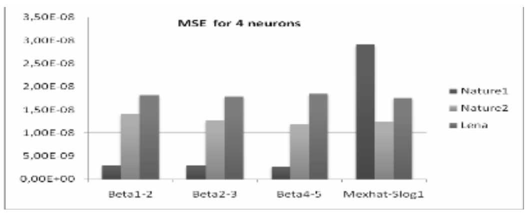


Fig. 7 – Variation of NMSE in term of wavelets library for Nature 1, Nature 2 and Lena, using 4 neurons and MLWNN architecture.

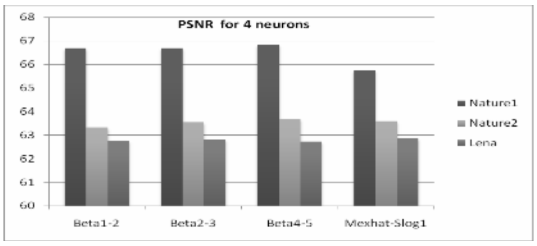


Fig. 8 – Variation of PSNR in term of wavelets library for Nature 1, Nature 2 and Lena, using 4 neurons and MLWNN architecture.

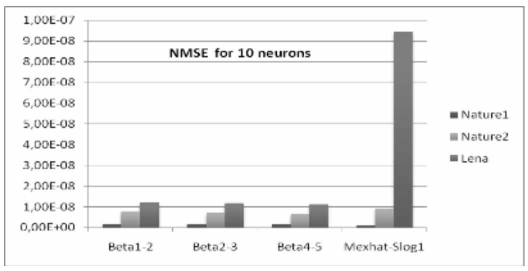


Fig. 9 – Variation of NMSE in term of wavelets library for Nature 1, Nature 2 and Lena, using 10 neurons and MLWNN architecture.

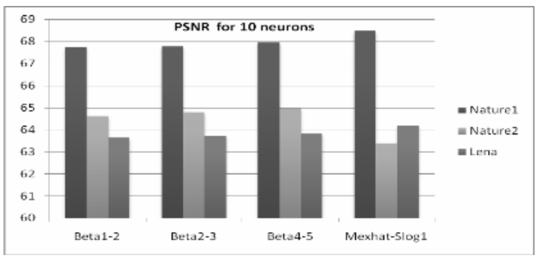


Fig. 10 – Variation of PSNR in term of wavelets library for Nature 1, Nature 2 and Lena, using 10 neurons and MLWNN architecture.

From these results (figure 7, 8, 9 and 10) we see that the NMSE depends on the wavelet library and the number of wavelets in the hidden layer. For example for nature1 we have an NMSE equal to 2.9929e-9 for a library constructed by Beta1 and Beta2 and for a net of 4 neurons in hidden layer where it is equal to 1.8244e-9 when we used a net of 10 neurons. Of course the time of compression is an essential factor: increasing the number of wavelets in hidden layer increases the time processing. For a multi library wavelet network based on selection procedure for initialization we can see that the difference of *NMSE* between 4 neurons and 10 neurons is small. To minimize time processing without biggest variation in *NMSE* we can reduce the number of neurons in hidden layer.

4.3 PERFORMANCES ACCORDING TO THE PSNR, AND NMSE FOR COLOR IMAGE COMPRESSION USING CLASSICAL WNN

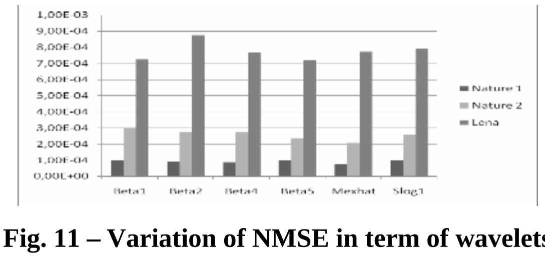


Fig. 11 – Variation of NMSE in term of wavelets library for Nature 1, Nature 2 and Lena, using 10 neurons and classical architecture.

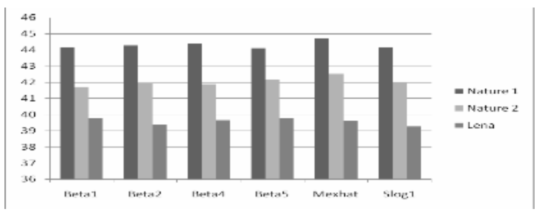


Fig. 12 – Variation of PSNR in term of wavelets library for Nature 1, Nature 2 and Lena, using 10 neurons and MLWNN architecture.

Comparing results given for the classical and proposed architecture (figure 11 and figure 12), we can conclude that the proposed architecture wavelet network, for the same value of compression ratio, gives results more interesting in term of mean square error. We can confirm that using more than one regressor (more than one wavelet in the library) in the wavelet library fit better the real output to the desired output.

5. CONCLUSION

Wavelet networks are a class of neural networks consisting of wavelets. In this paper, we have proposed a new Initialization by Selection algorithm for Multi library Wavelet Neural Network Training for the purpose of lossless color image compressing that provides improved efficiency compared to the classical wavelet neural networks.

From these results we can see the superiority of Beta wavelets family over the Mexican hat in the term of lossless color image compression.

We have shown that, when used a multi library wavelet networks and a selection procedure leads to results that are much more interesting than the classical architecture initialization.

Finally, although wavelet neural network approximation can be slow, we have shown that the loss in speed can be largely corrected by decreasing the number of wavelets in hidden layer without decreasing considerably the compression ratio and the PSNR.

As future research directions, we propose to use MLWNN in the case of adaptive self tuning PID controllers. The MLWNN is needed to learn the characteristics of the plant dynamic systems and make use of it to determine the future inputs that will minimize error performance index so as to compensate the PID controller parameters.

6. REFERENCES

[1] S. Mallat, A wavelet tour of signal processing. academic press 1998.
[2] Q. Zang, Wavelet Network in Nonparametric Estimation. IEEE Trans. Neural Networks, 1997. 8(2):227-236.
[3] Q. Zang et al., Wavelet networks. IEEE Trans.

Neural Networks, vol. 3, 1992. p. 889-898.

[4] H. Bourlard, Y. Kamp, Autoassociation by multilayer perceptrons and singular values decomposition, Biol. Cybernet. 1988. 291-294.

[5] A. Averbuch, D. Lazar, Image compression using wavelet transform and multiresolution decomposition, IEEE Trans. Image Process. 1996. p. 4-15.

[6] Hamdy S. Soliman, Mohammed Omari, A neural networks approach to image data compression, Applied Soft Computing, 2006. p. 258–271.

[7] G. Candotti, S. Carrato et al., Pyramidal multiresolution source coding for progressive sequences, IEEE Trans. Consumer Electronics, 1994. p. 789-795.

[8] S. Carrato, Neural networks for image compression, Neural Networks: Adv. and Appl. 2 ed., Gelenbe Pub, North-Holland, Amsterdam, 1992. p. 177-198.

[9] O.T.C. Chen et al., Image compression using self-organisation networks, IEEE Trans. Circuits Systems For Video Technol. 1994. p. 480-489.

[10] Slaven Marusic, Guang Deng, Adaptive prediction for lossless image compression, Signal Processing: Image Communication. 2002. p. 363–372.

[11] N.A. Laskaris, S. Fotopoulos, A novel training scheme for neural-network based vector quantizers and its application in image compression, Neurocomputing 2004. p.421-427.

[12] C. Foucher and G. Vaucher, Compression d’images et réseaux de neurones, revue Valgo n°01-02, Ardèche, 2001.

[13] Q. Zhang, Using Wavelet Network in Nonparametric Estimation, IEEE Trans. Neural Network, Vol. 8, 1997. p. 227-236.

[14] C. Aouiti, M.A Alimi, and A. Maalej, Genetic Designed Beta Basis Function Neural Networks for Multivariable Functions Approximation, Systems Analysis, Modeling, and Simulation, Special Issue on Advances in Control and Computer Engineering, vol. 42, no. 7, 2002. p. 975-1005.

[15] C. Ben Amar, W. Bellil and A. Alimi. Beta Function and its Derivatives: A New Wavelet Family. Transactions on Systems, Signals & Devices Volume 1, Number 3, 2005-2006. p. 275-293.

Wajdi BELLIL *was born in Gafsa, Tunisia, in 1977. He received his Master in Automatic and industrial Informatics from the National school of engineering, University of Sfax, Tunisia, in 2003. He is currently pursuing his doctoral degree in the Department of Electrical Engineering. His research interests are in the areas of wavelet neural network architecture sand learning algorithms, and applications of neural networks and machine learning techniques in image processing and pattern recognition.*

Chokri Ben Amar *was born in Sfax-Tunisia in 1964. he has been the studies director and the vice director of the National Engineering School of Sfax (ENIS) since June 2005. He has been a lecturer in Industrial Informatics at the department of Electrical Engineering in the National Engineering School of Sfax (ENIS) since 1999. He obtained the Master degree and the Doctorat at the Institut National des Sciences Appliquées de Lyon-France (INSA) in Industrial Automatics in 1990 and 1994. Since 1995, he has been an associate professor at the Ecole Nationale d’Ing´enieurs de Sfax. His research interests include multi-resolution image and video analysis, wavelets and neural Networks,with applications to function approximation and face recognition.*

Adel M. Alimi *was born in Sfax-Tunisia in 1966. He graduated in Electrical Engineering 1990, obtained a PhD and then an HDR both in Electrical Engineering in 1995 and 2000 respectively. Now, He is professor in Electrical & Computer Engineering at the University of Sfax. His research interest includes applications of intelligent methods (neural networks, fuzzy logic, genetic algorithms) to pattern recognition, robotic systems, vision systems, and industrial processes. He focuses his research on intelligent pattern recognition, learning, analysis and intelligent control of large scale complex systems. He is associate editor of the international journal: Pattern Recognition Letters. He was guest editor of several special issues of international journals (e.g. Fuzzy Sets & Systems, Soft Computing, Journal of Decision Systems, Integrated Computer Aided Engineering, Systems Analysis Modelling & Simulations). He was the general chairman of the International Conference on Machine Intelligence ACIDCA-ICMI’2005 & 2000. He is an IEEE senior member and member of IAPR, INNS and PRS.*

computing@computingonline.net
www.computingonline.net

ISSN 1727-6209
International Journal of Computing

FUZZY CLUSTERING METHODS IN MULTISPECTRAL SATELLITE IMAGE SEGMENTATION

Rauf Kh. Sadykhov [1], Valentin V. Ganchenko [1], Leonid P. Podenok [2]

[1] Computer Systems Department,
Belarusian State University of Informatics and Radioelecrtronics,
6 P. Brovka st, Minsk, Belarus
rsadykhov@bsuir.by, ganchenko@lsi-bas-net.by, http://www.bsuir.by

[2] Laboratory of System Identification,
United Institute of Informatics Problems,
National Academy of Sciences of Belarus,
6 Surganov st, Minsk, Belarus
podenok@lsi-bas-net.by, http://uiip.bas-net.by

Abstract: Segmentation method for subject processing the multi-spectral satellite images based on fuzzy clustering and preliminary non-linear filtering is represented. Three fuzzy clustering algorithms, namely Fuzzy C-means, Gustafson-Kessel, and Gath-Geva have been utilized. The experimental results obtained using these algorithms with and without preliminary nonlinear filtering to segment multi-spectral Landsat images have approved that segmentation based on fuzzy clustering provides good-looking discrimination of different land cover types. Implementations of Fuzzy C-means, Gustafson-Kessel, and Gath-Geva algorithms have got linear computational complexity depending on initial cluster amount and image size for single iteration step. They assume internal parallel implementation. The preliminary processing of source channels with nonlinear filter provides more clear cluster discrimination and has as a consequence more clear segment outlining…

Keywords: *Fuzzy Systems, Clustering, Image Segmentation, Multi-spectral Images, Satellite Images.*

1. INTRODUCTION

It is known that forests and wetland are the main factors preventing the decline in biodiversity on the Earth in aggressive conditions of human activity. The main problem is agricultural expansion and deforestation. Deforestation is the consequence of two main reasons – agricultural expansion and accidental events. But forestry and agriculture are inseparable and condemned to work hand in hand. Significant part of forest is damaged by fire, pests, irrational agricultural politics leading to change of ground water level and as result leads to sickness and wreck. At now it is possible to discriminate forest areas on early stage of damaging using multi-spectral images of high spatial resolution received from satellites. That technology started about 40 years ago to monitor Earth surface at now is the effective instrument of ecological and agricultural monitoring such the regions as forests and wetland and preventing any accidents. Multi-spectral satellite images are able to bring us information in both visible and invisible spectral bands about vegetation, water temperature and land cover.

Multi-dimensional cluster analysis and segmentation are base procedures in thematic processing the multi-spectral images received from remote sensing satellites. There are lot of clustering and segmentation methods which have different benefits and imperfections. The special class of such the methods is represented by fuzzy clustering ones.

2. METHOD DESCRIPTION

Three clustering algorithms based on fuzzy methods where developed and utilized as part of segmentation software. There are fuzzy c-means [1] (FCM) and its variants – Gustafson-Kessel clustering algorithm [2, 3] and Gath-Geva one [4].

FCM is a method of data clustering which allows one data objects to be a member of two or more clusters. This method developed by [5] and improved by [6] is based on minimization of the following objective function:

$$J = \sum_{i=1}^{N} \sum_{j=1}^{C} \mu_{ij}^{m} \parallel x_i - s_j \parallel, \qquad (1)$$

where m is real number ≥ 0, μ_{ij} – degree of membership of xi in cluster s_j , x_i is the multi-dimensional data object, s_j is the multi-dimensional center of the cluster, and $\|.\|$ is any norm expressing the similarity between any measured data and the cluster center. Fuzzy clustering is carried out through an iterative optimization of the J_m with the update of membership matrix μ_{ij} and the cluster centers sj using following algorithm:

1. The data set $X = (x_j) = (x_{j1}, x_{j2} \ldots, x_{jp})^T$ given. We choose the number of clusters, denoted by "c", 1 < c < n, where p – dimension of data set.
2. Initialize the partition matrix μij using random number generator in range [0, 1]:

$$\sum_{i=1}^{c} \mu_{ij} = 1 \qquad (2)$$

3. Calculate the cluster centers $s_i = (s_{i1}, s_{i2}, \ldots, s_{ip})^T$ (i = 1, 2, ..., c):

$$s_i = \sum_{j=1}^{n} \mu_{ij}^2 x_j \bigg/ \sum_{j=1}^{n} \mu_{ij}^2 \qquad (3)$$

4. Calculate the error:

$$E = \sum_{i=1}^{c} \sum_{j=1}^{n} \mu_{ij}^2 (s_i - x_j)^T (s_i - x_j) \qquad (4)$$

If error is small enough, then stop iterations, else go to item 5.

5. Calculate the "new" partition matrix μ_{ij}, where d_{mj}^2 is similarity measure:

$$\mu_{ij} = \left(\sum_{m=1}^{n} d_{ij}^2 / d_{mj}^2 \right)^{-\frac{1}{2}} \qquad (5)$$

and then go to step 3.

Fuzzy c-means, Gustafson-Kessel, and Gath-Geva clustering algorithms are distinguished in the definition of distance function between the objects to be classified: Fuzzy C-means:

$$d_{ij}^2 = (s_i - x_j)^T (s_i - x_j) \qquad (6)$$

Simple Euclidean distance provides hyper-spherical form of clusters. Gustafson-Kessel (relation (7)) and Gath-Geva (relation (8)) provide ellipsoidal form of clusters and more comprehensive partitioning the multi-dimensional data.

$$d_{ij}^2 = (s_i - x_j)^T (det(F_i)^{\frac{1}{2}} F_i^{-1})(s_i - x_j) \quad (7)$$

$$d_{ij}^2 = \frac{det(F_i)^{\frac{1}{2}}}{\alpha_i} \exp(\frac{1}{2}(s_i - x_j)^T (F_i^{-1})(s_i - x_j)) \quad (8)$$

F_i is covariance matrix and α_i is calculated in following manner:

$$\alpha_i = \frac{1}{n} \sum_{j=1}^{n} \mu_{ij} \qquad (9)$$

Covariance matrix is used to form non-spherical clusters which are more suitable for multi-dimensional data partitioning. That also leads to visible difference in algorithm convergence, effectivity and performance of processing the multi-spectral images.

Distance function for Gustafson-Kessel and Gath-Geva algorithms uses covariance matrix to take into account different metrics (scales) in different dimensions. The large images to be processed might be the reason of weak covariance matrix conditionality that results in heavy losses of aptitude for discrimination for Gustafson-Kessel and overflows for Gath-Geva when fixed digit capacity arithmetic based on atomic numerical types is used. To keep of those problems standard normalization methods and arbitrary (multi-precision) arithmetic are the sufficient efforts. When d_{ij}^2 becomes small the overflow may occur. To prevent that case the distant function is confined from below by some smallest value d_{min}.

Due to source signal noise the some spatial segment granularity occurs. To reduce that granularity the weak nonlinear filtering using algorithm [7] was applied to source channels. That algorithm may be used both for edges extraction and for nonlinear filtering. It does not lower the sharpness of transitions of channel brightness the boundaries of cover types remain clear. As a result the boundaries of spatial segments after clustering process also remain legible. Algorithm works with values of brightness and pixels coordinates simultaneously. To form the homogeneous areas or search the edges on the gray-scale image a round mask is used. Usually the mask's radius is 3.4 pixels which gives mask of 37 pixels size. The mask is placed at each point of the image and the brightness of each pixel of the mask is compared with the brightness of the mask central pixel (relation (10)), where I(r0) – the brightness value of the mask's center, I(r) – the brightness value of the mask pixel, t specific threshold.

$$c(\mathbf{r}, \mathbf{r}_0) = \begin{cases} 1, I(\mathbf{r}) - I(\mathbf{r}_0) \le t \\ 0, I(\mathbf{r}) - I(\mathbf{r}_0) > t \end{cases} \quad (20)$$

$$n(\mathbf{r}_0) = \sum_{\mathbf{r}} c(\mathbf{r}, \mathbf{r}_0) \quad (31)$$

The result of the comparisons is conform to the relation (11), where n is the quantity of pixels in the USAN (Univalue Segment Assimilating Nucleus). Then that sum should be minimized, so the algorithm is called SUSAN (Smallest USAN). Parameter t means the maximum of ignored noise. Then n is compared with it's thresholding value g, which is $3n_{max}/4$, where n_{max} – maximum value which could be assigned to n.

3. EXPERIMENTAL RESULTS

To test the fuzzy clustering algorithms the multi-spectral Landsat images have been used. Fig. 1 shows channels 2-5 which were chosen for processing. Landsat channels shown on fig.1 have different dispersion and were equalized using `netpbm` utilities to make them visible more clearly. The channel data to be fed into clustering software remain untouched because of clustering algorithm takes in to account real distribution of channel signal level on one's own. Raw output data of developed segmentation software is 2-d vector field over R^N, where N is a number of clusters. Each component of that field is strictly increasing function of probability for pixel to be member of one of N clusters.

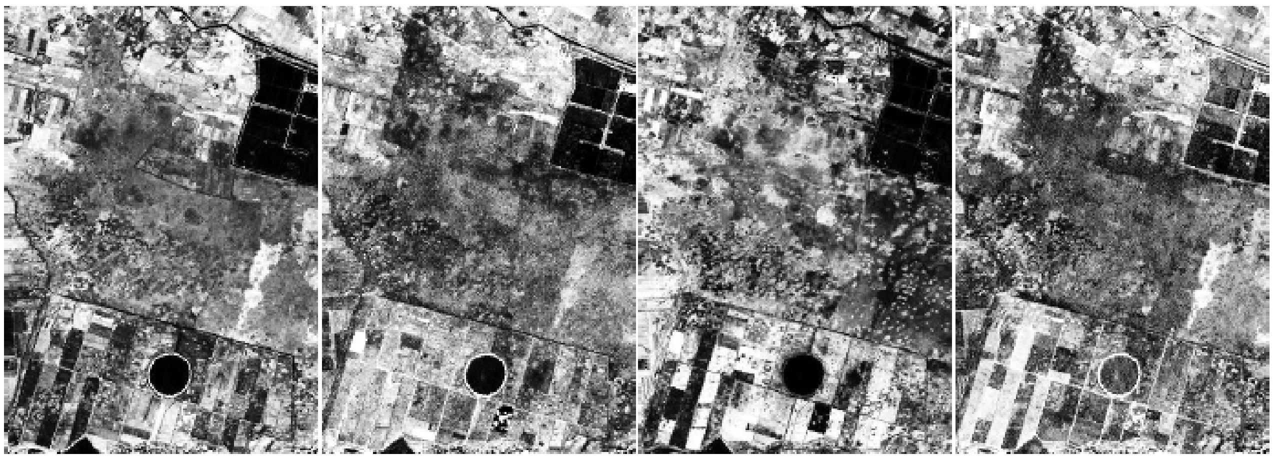


Fig. 1 – Channels 2-5 of multi-spectral Landsat image to be processed with fuzzy clustering algorithm.

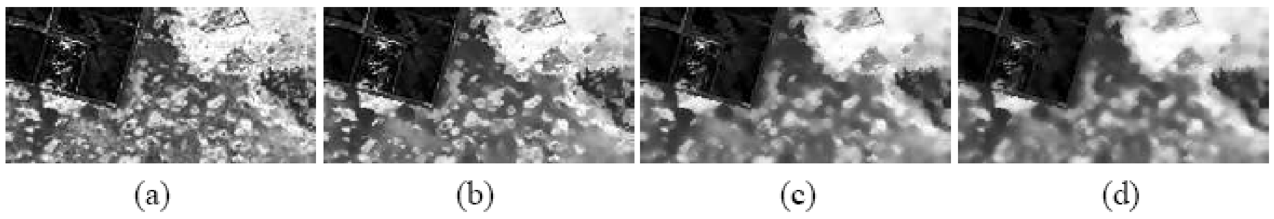


Fig. 2 – Nonlinear filtering samples: (a) original image, (b) brightness threshold = 7, (c) brightness threshold = 15, (d) brightness threshold = 20.

In order to obtain exploitable data for a classification scheme, we first needed to extract relevant information of raw Nomarski's microscopy issued images. We proposed to proceed in two steps [2]: first a detected items' images extraction phase and then an appropriated coding of the extracted images.

On presented images one can see some well discriminated land cover types. There are open water, forests, wetlands, bushes and agriculture areas. Results of segmentation on 15 clusters using fuzzy C-means without any preliminary filtering show that there also are fuzzy boundaries of segments on the cluster map going on to sand in some places. To diminish such the phenomena one can use filter which will smooth slightly changing areas rather blur clearly discriminated land cover type edges. Most appropriate filter having got such the behaviour is the non-linear one described in [7]. The fig. 2 demonstrates smoothing properties of that filter when brightness threshold was chosen be equal to 7, 15, and 20. Besides the brightness threshold parameter the filter has spatial one – distance threshold which was set into default value corresponding to radius equal to about 3.4 and was not changed across experiments. As one can see, smoothing property of that filter increases with brightness threshold grows. At the same time some part of edges remains sharp. Amount of sharp edges and consequently cluster granularity depend on brightness threshold. Thus we get additional degree of freedom in segmentation process control that could dramatically improve the segmentation results.

Nonlinear filtering lowers intra-class covariance but practically doesn't touch interclass ones, so cluster discrimination doesn't suffer. Smoothing the channels before segmentation allows to eliminate fine granularity of output segment map and to merge the small cluster with large ones. In reality this means disappearance of some real small objects, for example, bushes in wetland areas. Nevertheless, sometimes it is necessary to get generalized segment map. The SUSAN filter allows fulfilling nonlinear processing varying the threshold in wide region achieving the smoothing results from negligible small up to very strong.

Fig. 3-6 demonstrate examples of segmentation using non-linear filtering with different brightness threshold for Gustafson-Kessel and Gath-Geva clustering algorithms. The cluster number was chosen taking into account real diversity of land covers for territory being investigated. Actually a lot of experiments with different cluster number were fulfilled. As soon as segmentation become to look stable cluster number was stated.

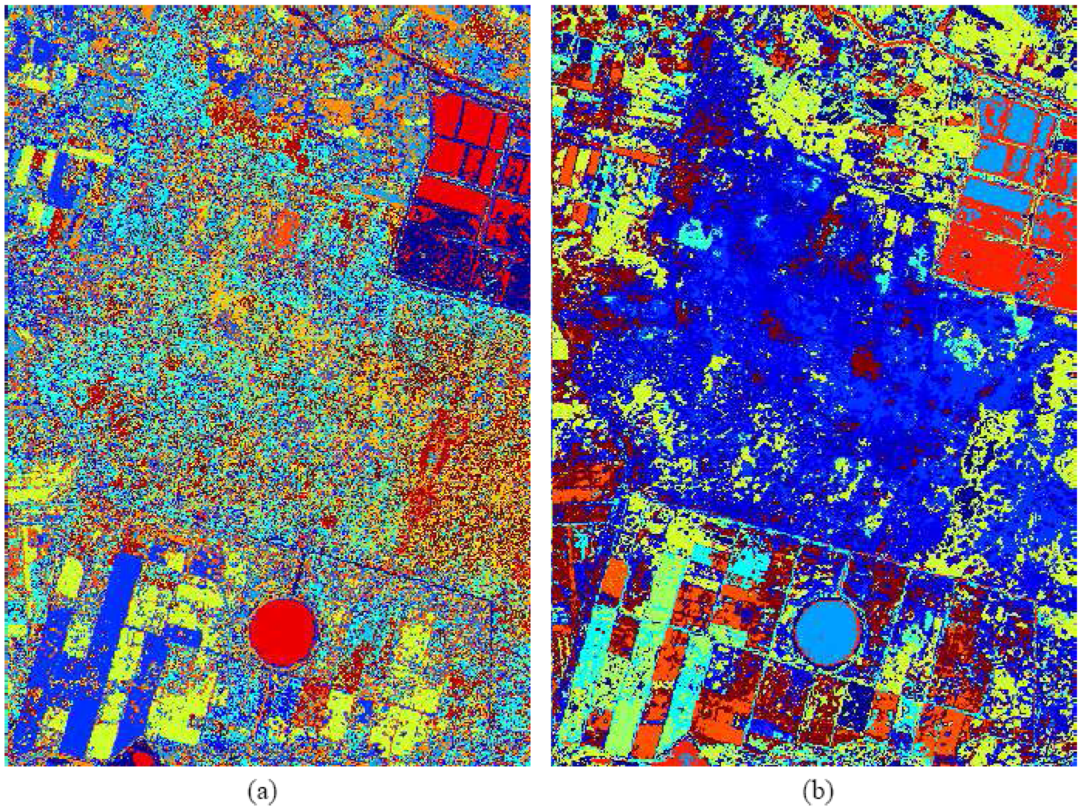


Fig. 3 – Segmentation results using Gustafson-Kessel (a) and Gath-Geva (b) algorithms at 20 clusters without any filtering.

Obtained results have transformed into scalar field using simple maximal probability solver to get the possibility of visual evaluation and have presented as colour map of spatial partitions. As a result every pixel became a member of one cluster. All pixels of same colour are members of same cluster. No any intelligent algorithm was used to colourize segment map so the same areas on different pictures have different colours. As one can see the growth of the threshold leads to raising the segment area when target number of clusters is fixed. However it should be pointed that general structure of segment arrangement and its structure in relation with land cover types is not significantly changed.

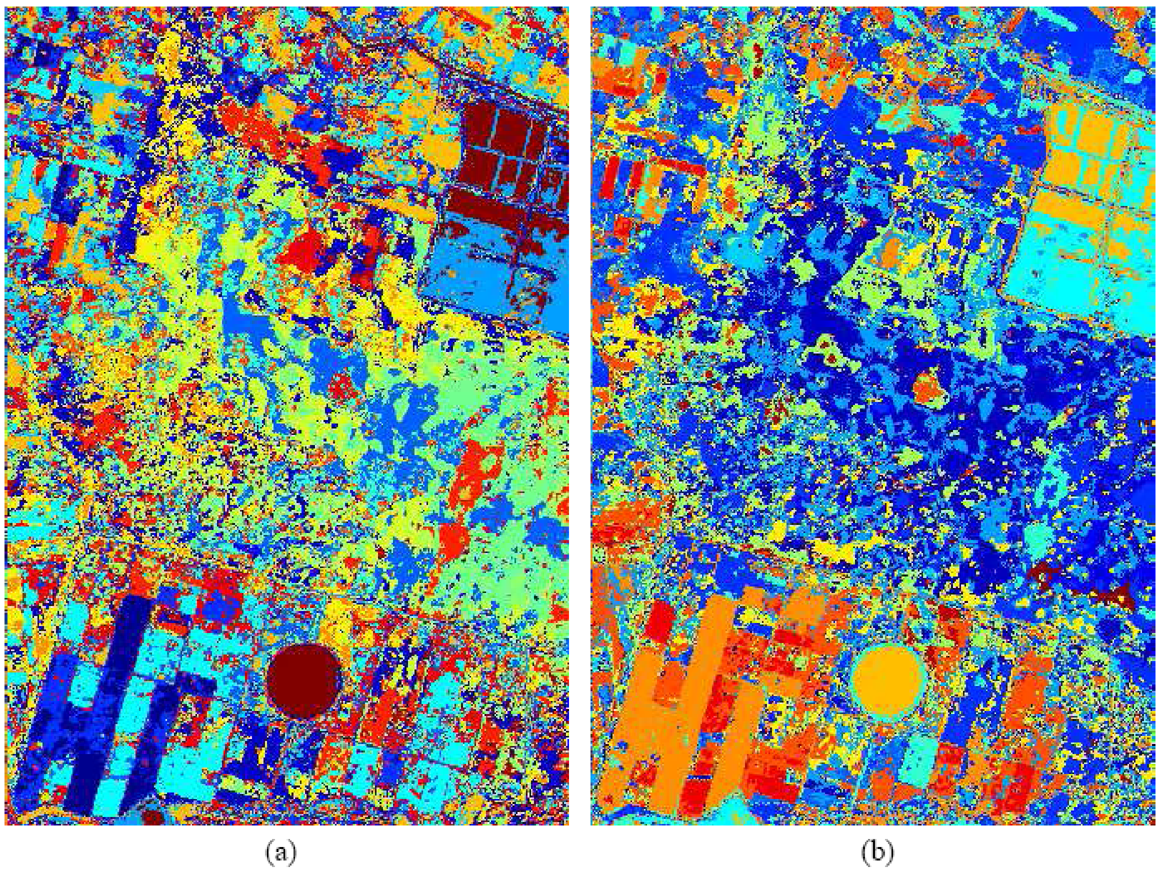


Fig. 4 – Segmentation results using Gustafson-Kessel (a) and Gath-Geva (b) algorithms at 20 clusters using filtering with threshold = 7.

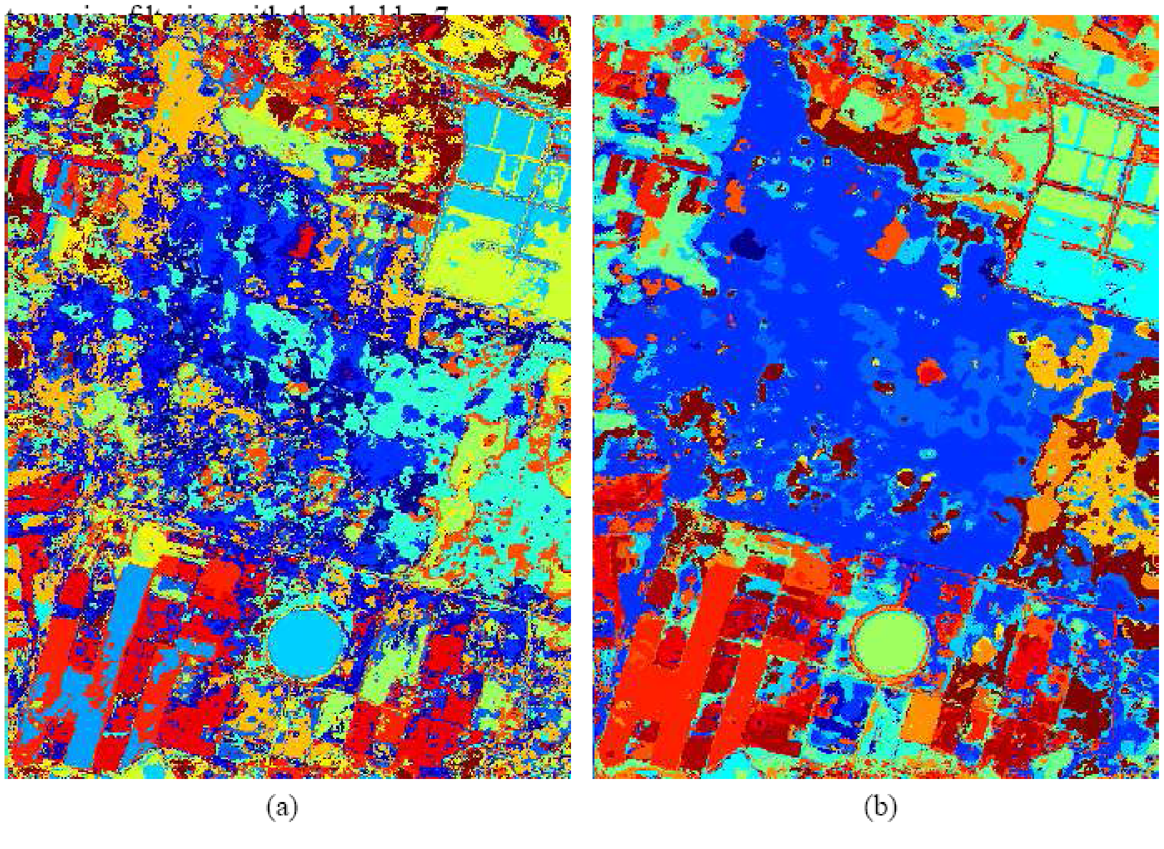


Fig. 5 – Segmentation results using Gustafson-Kessel (a) and Gath-Geva (b) algorithms at 20 clusters using filtering with threshold = 15.

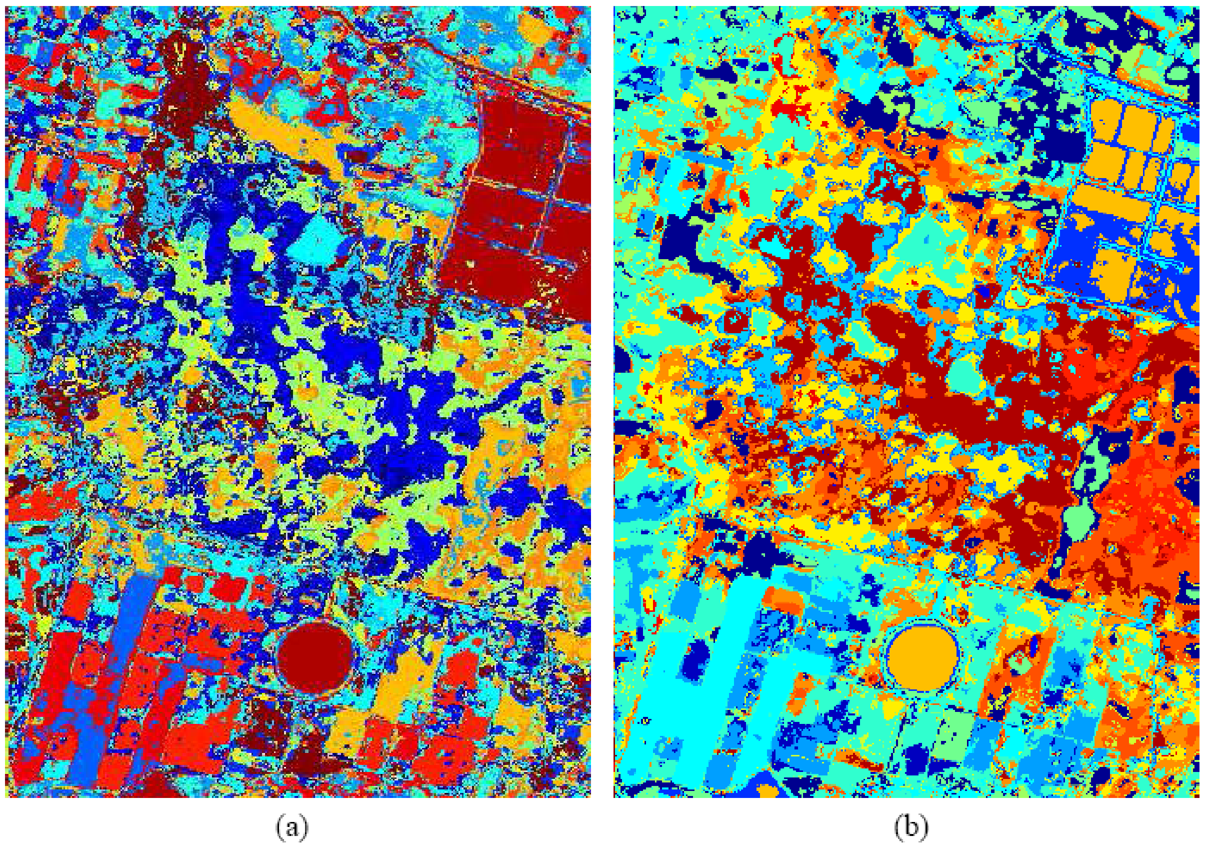


Fig. 6 – Segmentation results using Gustafson-Kessel (a) and Gath-Geva (b) algorithms at 20 clusters using filtering with threshold = 20.

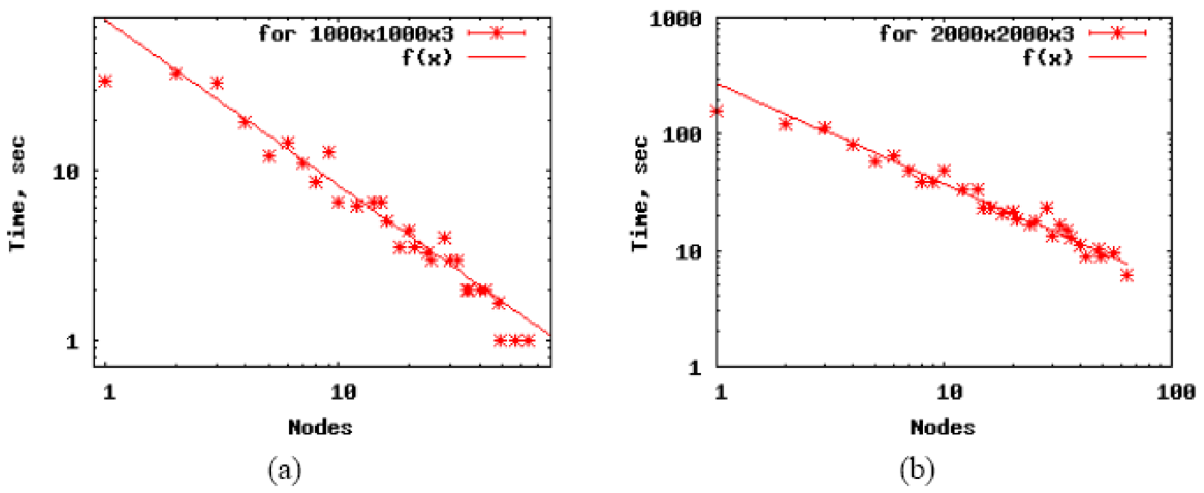


Fig. 7 – Dependence of processing time from node count for colour images with size of 2000x2000 (a) and 1000x1000 (b) pixels.

4. ACCELERATION OF CALCULATIONS

For increase in usability and calculation efficiency can be made with use in systems of mass parallelism, and also have connection with GIS. As a basis for parallel processing interface MPI (The Message Passing Interface) is chosen. Tests of system of processing of images were made on a supercomputer "SKIF K-1000". At carrying out of experiments it was involved from 1 up to 64 computing units for colour images in the size 2000x2000 and 1000x1000 pixels. Dependence of a system operating time of on amount of the involved computing units is resulted on the schedules resulted on fig. 7.

5. CONCLUSION

Segmentation results of multispectral Landsat images obtained using fuzzy clustering methods such as Fuzzy C-means, Gustafson-Kessel, and Gath-Geva with and without preliminary nonlinear filtering testify that segmentation using fuzzy clustering methods provides good-looking discrimination of land cover types that occurs in uch the complex cases as wetland, water-meadow, and bush areas. The discrimination quality of segmented images was tested and approved by land-improvement specialists using data of land-based expedition. The non-linear filtering with guided smoothing is the convenient instrument of preliminary processing for semi-automatic segmentation of complex land covers.

Implementations of Fuzzy C-means and Gustafson-Kessel algorithms have got linear computational complexity depending on innitial cluster amount and image size for single iteration step. All the algorithms assume internal parallel implementation for MPP computer. The preliminary processing of source channels with nonlinear filter provides clearer cluster discrimination and has as a consequence more clear segment outlining and provides operated generalization of output segment maps.

Really, segmentation software returns the results as 2-d vector field v_i, which is the strictly increasing function of probability p_i that pixel is a member of i-th cluster. When using more complex tool then simple maximal probability solver, for example, maximal likehood one, it is possible to significantly improve the results of segmentation and to avoid large part of manual processing the multi-spectral data.

6. REFERENCES

[1] Hoppner, F., Klawonn F., Kruse, R., Runkler, T.: Fuzzy Cluster Analysis. Wiley, Chichester (1999).

[2] Gustafson, D. E., Kessel, W. C.: Fuzzy clustering with fuzzy covariance matrix. In Proceedings of the IEEE CDC, INSTICC Press, San Diego (1979) 761–766.

[3] Babuska, R., van der Veen, P. J., Kaymak, U.:, Improved covariance estimation for Gustafson-Kessel clustering. IEEE International Conference on Fuzzy Systems (2002) 1081–1085".

[4] Gath, I., Geva, A. B.: Unsupervised optimal fuzzy clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence (1989) 7:773–781.

[5] Dunn, J. C.: A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters. Journal of Cybernetics (1973) 3: 32-57.

[6] Bezdek, J. C.: Pattern Recognition with Fuzzy Objective Function Algoritms. Plenum Press, New York (1981).

[7] Smith, S. M., Brady, J. M.: SUSAN – a new approach to low level image processing. International Journal of Computer Vision May (1997) 23(1):45-78.

__Rauf Kh. Sadykhov__, in 1967 graduated from Azerbaijan Polytechnic Institute (Baku) on the specialty "Mathematical and Computing the Instruments and Devices". After graduation from the Institute he attended the postgraduate course at the Institute of Engineering Cybernetics in Minsk. In 1991 he defended his thesis for a scientific degree of a doctor of engineering science in the field of computing science and in 1992 has obtained a professor.s scientific rank.

Since 1995 R.Kh. Sadykhov is a head of Computer System Department in Belarusian State University of Informatics and Radioelectronics and simultaneously he is a head of System Identification laboratory, Institute of Engineering cybernetics of the Belarusian Academy of Sciences. R.Kh. Sadykhov has published more than 350 scientific works, including books, patents, papers and reports at the International Conferences, Symposiums and Workshop.

The area of the scientific investigations includes: digital signal and image processing, recognition of handwritten symbol and signature identification, remote-sensing object recognition, computer vision system for the control and recognition, intellectual neural systems, multi-agent systems, parallel architectures for digital signal and image processing. Prof. R. Sadykhov is the vice-chairman of Belarusian Association of Pattern Recognition (IAPR) and Belarus SIG of International Neural Network Society (INNS), the member of IEE (United Kingdom).

__Valentin V. Ganchenko__ in 2006 graduated from Belarusian State University of Informatics and Radioelectronics (Minsk) on the specialty "Computers, Systems and Networks". After graduation from the University he attended the postgraduate course at the "United Institute of Informatics Problems of the National Academy of Sciences of Belarus" (UIIP NAS of Belarus) in Minsk.

Since 2006 Valentin V. Ganchenko is a collaborator of the Laboratory of System Identification in United Institute of Informatics Problems of the National Academy of Sciences of Belarus.

The area of the scientific investigations includes: digital image processing, remote-sensing object recognition, parallel architectures for digital image processing.

***Leonid P. Podenok** in 1982 graduated from Belarusian State University (Minsk) on the specialty Physics.*

Leonid P. Podenok is a collaborator of the Laboratory of System Identification in United Institute of Informatics Problems of the National Academy of Sciences of Belarus.

The area of the scientific investigations includes: multispectral satellite image processing, parallel processing, space telemetry processing, public key cryptography and fast arithmetics.