APPLICATION OF MODIFIED SOM NEURAL NETWORKS ON ACYCLIC DATA STRUCTURES

Gabriela ANDREJKOVÁ, Jozef ORAVEC
Institute of Computer Science, Faculty of Science, Pavol J. Šafárik University in Košice,
Jesenná 5, 042 00 Košice, Slovak Republic, tel.: +421 55 234 6228, e-mail: gabriela.andrejkova@upjs.sk

ABSTRACT

Graphs as data structures are used in many applications, for example image analysis, scene description, natural language processing. The paper deals with Acyclic Graph Data Structures (AGDS) and with a learning process in a model of a Self-Organizing Map (SOM) that has been modified for processing of AGDS. To the modified SOM Neural Network (SOM NN), there are added contexts and counters which are built in a training phase of the neural network. The trained SOM NN in active phase can compute information which is used to built some acyclic substructure. The prepared application was tested on the real data of study programs, the test results are evaluated using a winner differentiation and a confidence criterion.

Keywords: *Acyclic data structure, self-organizing map, counter propagation algorithm, learning algorithm*

1. INTRODUCTION

Graphs are very important models of data structures, they give possibilities to represent facts in vertices and relations among facts in edges. In some applications, the structured data are often connected to a learning process from examples. The graph $G = (V,E)$, where V is the finite set of vertices (facts) $|V| = n$ and E is the set of edges (relations) $|E| \leq n^2$ should be represented by a matrix of type $n \times n$. Problems can be seen here are the following: for each example (presented by some graph) used in the training procedure we can have a different type of the matrix; if the biggest type of the matrix is used for all graphs then there are too many free positions in the matrix, which do not represent anything (any relation). It means, it is important to find some other representations of graphs (data structures) and it is necessary to do it in a connection to a model of computation.

SOM NN have been proposed by many authors [2], [3], [4], [8], [12], as models which are very good for a learning of graph data structures. It means, the graph structures in connection to neural networks should be represented in some special representation (a trained neural network) and the neural networks can be trained to graph structures. In the training, it is possible to prepare input data of the graph in some parts, it is not necessary to put a full graph as one input data to the network. The shape of a output substructure is very important in many solved problems. In [1], we have prepared theoretical points of view to a presented application of using SOM neural networks to input and output acyclic data structures.

The paper is prepared in the following sections. In the first section we describe the motivation to the study of the problem, the second section is oriented to a description of acyclic data structures and to a description of formal problems solved in the paper, in the third section we describe the model of prepared neural network - Counter Propagation SOM for Structured Data (CP SOM SD). The next section contains a description of a data encoding and a training algorithm of modified SOM NN. In the last section, there are defined criteria of an evaluation of CP SOM SD and evaluated the prepared solution in some examples. In the conclusion, we summarize the work which was done and describe some plan for the following work.

2. MOTIVATION - PREREQUISITES IN STUDY PROGRAMS

Many study programs at universities are built in a system of three subject types: compulsory subjects, compulsory elective and elective subjects. From the theoretical point of view, all subjects built a partially ordered set according to the order of subject passings.

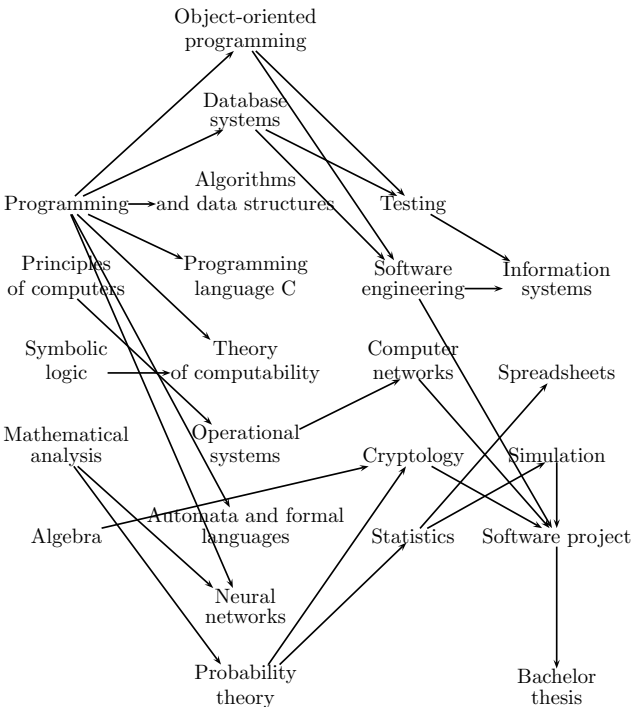

Fig. 1 Mapping of prerequisites for study program in informatics

The partial ordering is defined by prerequisites of subjects. The part of study program in informatics at Faculty of Science of P. J. Šafárik university in Košice is mapped in the Fig. 1.

At the beginning of academic year, the students choose new subjects which they will study in the current academic year. They register the chosen subjects in an Academic information system. For example, a registration of the subject *Software engineering* can be taken after passing subjects *Object-oriented programming* and *Database systems*.

The main problem: The system has information about the structure of all subjects in the study program (some acyclic graph structure). The information about passed subjects of the student can be put to the system in two ways:

- The information is in the list of passed subjects in Academic information system. It is possible to obtain the acyclic structure of all student's passed subjects in automatic way.
- The way is not automatic. The student will give only some passed subjects. We will follow this way.

When the student gives some passed subjects and he chooses some subject, which he try to study for the next academic year, the system should answer him automatically *NO (you can not to register the subject, you have not fulfilled prerequisites)* in real case or it can answer *YES (it is O.K.)*. Or it is possible to ask the system for some recommended subjects which the student can study in the following academic year.

3. ACYCLIC GRAPH DATA STRUCTURES AND TWO SOLVED PROBLEMS

A directed graph or digraph is a pair $G = (V,E)$, where V is a set, whose elements are called vertices or nodes, E is a set of ordered pairs of vertices, called directed edges.

A Directed Acyclic Graph (DAG), is a directed graph with no directed cycles. It is construct by a collection of vertices and directed edges, each edge is connecting from one vertex to another, such that there is no way to start at some vertex v and follow a sequence of edges that eventually loops back to v again. In the DAG graph, there exists one vertex at least without input edges.

Let $G = (V,E)$ be a DAG graph, V_s be a set of all vertices without input edges. We call the elements in V_s as **sources**. Some vertices in the DAG graph have some special marks. Let V_m be a set of all marked vertices.

3.1. The first problem - a reachable vertex

Let $v^z \in V$ be a chosen vertex. Let $G_p = (V_p, E_p)$ be a graph of all predecessors of the vertex v^z of all levels (predecessors of predecessors, ...). (We construct all paths from sources to v^z.) Is it $V_p \subseteq V_s$?

3.2. The second problem - a set of successors

Let $G_t = (V_t, E_t)$ is a graph of successors of all vertices in V_s of all levels which belong to V_m too. (We construct all longest paths from sources using vertices from V_m, all vertices in the longest paths belong to V_t.) $V_t \subseteq V_m$. Let V_t^l be the set of the last vertices in the longest paths from sources. The problem is to find a set V_f of vertices - successors in the first level of vertices in V_t^l?

3.3. Examples

We used the DAG to define the study program. The subjects are in the vertices and the oriented edges represent prerequisites. If the subject A is the prerequisite of the subject B then in DAG is the oriented edge from A to B. The subject A is called **right prerequisite** or the **first level prerequisite** to B. If C is the right prerequisite to A, the C is the **second level prerequisite** to B.

If we choose some vertices from DAG together with the oriented edges among them then we get a set of graphs (one or more subgraphs) and the previous information in the DAG is reduced. The reduced information is sometimes enough to compute results, but in more cases it is not enough.

Some study program is constructed as a DAG graph. The marked vertices in V_m represent passed subjects of some student. The formal problems should be represented in the following way:

- The first problem: Can the student register a subject v^z?
- The second problem: Construction of the set of subjects which can be register by student, if he passed subjects in V_m.

4. MODEL OF CP SOM SD

We describe a model of a neural network which has been used for the computation of the answers to given queries about passing subjects. Our model was motivated by the model of the neural network working with tree structured data [6], [10], [9].

4.1. SOM for Tree Structure

The SOM for Structured Data (SOM SD) has been described for processing of labeled trees with fixed fan-out k [5], [11]. We describe very shortly the model which has been used for the classification of artificially constructed pictures represented by tree structured data according to [6], [3].

It is assumed that the neurons are arranged in rectangular d-dimensional lattice structure. Each neuron can be enumerated by tuples $\bar{i} = (i_1, \ldots, i_d)$ in $\{1, \ldots, N_1\} \times \cdots \times \{1, \ldots, N_d\}$ where $N_i \in \mathcal{N}$. The vector $\bar{i}$ describes the position of the neuron in the lattice. Let $N = N_1 * \cdots * N_d$, N be the number neurons in the lattice, n is a dimension of an input. Each neuron $n_{\bar{i}}$ is equipped with a weight $w_{\bar{i}}$ and k context vectors $c_1^{\bar{i}}, \ldots, c_k^{\bar{i}}$ in $\mathscr{R}^d$. They represent the contexts of proceed tree given by the k subtrees and the indexes of their respective winners. The winning neuron with index $I(t)$ to given a tree t with root label a and subtrees $t_1, \ldots, t_k$ is recursively defined by

$$I(\varepsilon) = (-1, \ldots, -1), \varepsilon \text{ - empty tree}$$

$$I(t) = I(a(t_1, \ldots, t_k)) = arg\,min_{\bar{i}}\{\bar{i}|\alpha \cdot \|a - w_{\bar{i}}\|^2 + \beta \cdot \|I(t_1) - c_1^{\bar{i}}\|^2 + \cdots + \beta \cdot \|I(t_k) - c_k^{\bar{i}}\|^2\}.$$

$\alpha, \beta > 0$ are parameters to control mediation between the amount of pattern match versus context match. The empty

tree ε is represented by an index not contained in the lattice, $(-1,\ldots,-1)$. The weights of the neurons consist of compact representations of trees with prototypical labels and prototypical winner indices which represent the subtrees. Starting at the leaves, the winner is recursively computed for the entire tree. The index $\bar{i}$ of the winner for the subtrees is computed. The attached weights $(w_{\bar{i}}, c_1^{\bar{i}}, \ldots, c_k^{\bar{i}})$ are moved into the direction $(a, I(t_1), \ldots, I(t_k))$ after each recursive processing step, where $I(t_i)$ denotes the winning index of the subtree t_i of the currently processed part of the tree. The weights of neighbor neurons are updated in the same direction with smaller learning rate.

In [4], it is described SOM SD in a supervised approach. The training set is defined as $T = \{(D_i, d_i), i = 1,2,\ldots,m\}$ where D_i is the data structure with desired value d_i, the input vector for each node of this data structure is extended by adding a new field containing information about target. The model is called sSOM SD and it was applied on a classification task with 12 classes.

4.2. Model CP SOM for Acyclic Data Structures

The developed model is drawn in the picture Fig. 2. The described model can be divided into three part:

A. **The lattice of neurons, contexts and counters - CP SOM SD**. The network (in the middle part of Fig. 2) is trained for the acyclic structure of the study program. In the process, there are built contexts for prerequisites of subjects and counters for all prerequisites of a given subject.

The contexts are $r_x^1, r_y^1, r_x^2, r_y^2, \ldots, r_x^k, r_y^k$. The weights $\mathbf{c_1}, \mathbf{c_2}, \ldots, \mathbf{c_k}$ are the weights to contexts. The counters are $u_1, u_2, \ldots u_K$ and each counter contains all prerequisites of one subject.

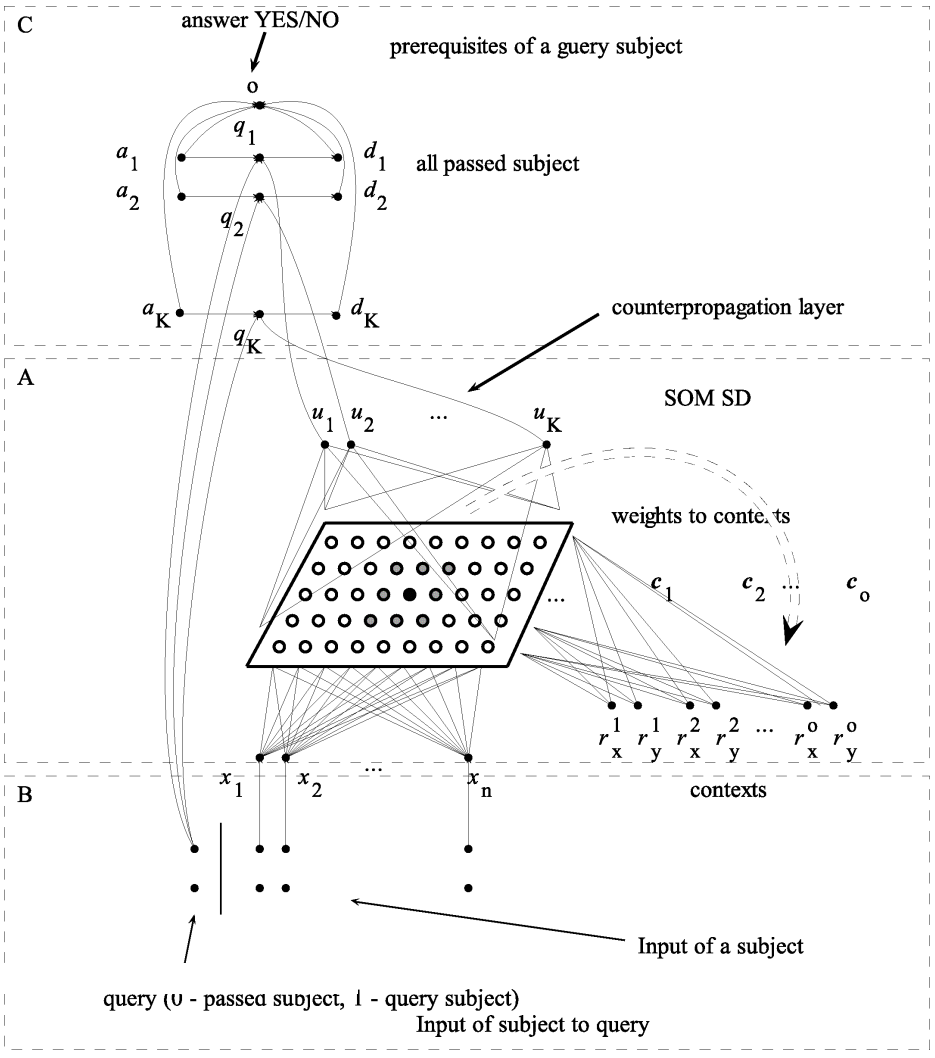

Fig. 2 The structure of CP SOM neural network

B. **Input part** - vector $\mathbf{x} = (x_1, \ldots x_n)$ is important in the training of the network for the acyclic data structure of the study program and in the active phase of computation, it is necessary to add one bit with information about passing the subject.

C. **Output part** is built from the vectors $\overline{a} = (a_1, a_2, \ldots, a_K), \overline{q} = (q_1, q_2, \ldots, q_K)$ and $\overline{d} = (d_1, d_2, \ldots, d_K)$. The $(0,1)$ - vector $\overline{a}$ represents the passed subjects given on input (given by student), the $(0,1)$ - vector $\overline{d}$ explains if the subject has prerequisites or not. From the vectors $\overline{a}, \overline{q}$ and $\overline{d}$ is computed result if the student can register the queried subject. The vector $\overline{q}$ is computed using counters and one bit information in the input.

5. ENCODING OF DATA STRUCTURES AND THE TRAINING OF THE NEURAL NETWORK

5.1. Encoding of training structures and training sets

Let $K = |V|$ be the number of subjects in the acyclic data structure of the study program. The subjects are encoded in the binary alphabet $\{0,1\}$, it means each subject $\mathbf{p^i} = [p_1^i, p_2^i, \ldots, p_b^i] \in \{0,1\}^b, 1 \leq i \leq K$, where $b = log_2(2 * K + 10000)1$ is the numbers of bits used to the encoding of each subject.

The preparation of training structures — to each subject is prepared a substructure with all prerequisites on the first and second level. The prepared substructures have some the same vertices from the structure.

Let $\mathbf{T_c}$ be training set to the set of ADS S.

$$\mathbf{T}_c = \{(D_l, d_l), D_l \in S, d_l \in \{0,1\}^k, l = 1, \ldots, p)\}\} \quad (1)$$

where p is the number of training patterns (the number of patterns is the same as the number of all vertices in all training data structures) and d_l is an expected value to a data structure D_l.

5.2. Training of the neural network

The distance of input $\mathbf{x}$ from the i-th neuron in the lattice and in connection to the contexts was computed according the formula (2). In the formula, there is the second part where the distances from contexts $(\mathbf{r^j}, \mathbf{c_i^{(j)}})$ are evaluated and a sum of them have an influence to the distance value.

$$d_i = \alpha \|\mathbf{x} - \mathbf{w}_i\|^2 + \beta \left(\left\| \mathbf{r}^1 - \mathbf{c}_i^{(1)} \right\|^2 + \ldots + \left\| \mathbf{r}^k - \mathbf{c}_i^{(k)} \right\|^2 \right), \quad (2)$$

where $\alpha, \beta > 0$ are some parameters.

Let the winner is i^*, it means winner is one of the closest neurons to the input $\mathbf{x}$. It is computed by 3

$$i^* = arg\,min_{j=1,\ldots,N} ||\mathbf{x} - \mathbf{w_j}|| \quad (3)$$

The adaptation of weights in the lattice is done according the formula (4)

$$\triangle \mathbf{w}_j = \eta(t) h_{i^*}(j,t) \left\| \mathbf{x} - \mathbf{w}_j \right\|^2, \quad (4)$$

where $0 < \eta(t) < 1$ is a learning rate, $h_{i^*}(j,t) = e^{-\frac{dist(i^*,j)^2}{2\sigma^2(t)}}$, and $dist(i^*, j)$ is a distance of j-th neuron to the winner neuron i^*. The output layer contains K Grossberg's units with weights $\mathbf{u}_i, i = 1, \ldots, K$. The counter propagation was applied for training $\mathbf{u}_i$.

$$\mathbf{u}_r^{(t)} = \mathbf{u}_r^{(t+1)} + \eta(t)(-\mathbf{u}_r^{(t-1)} + \mathbf{d}_r^{(t)})\mathbf{z} \quad (5)$$

where $\mathbf{z}$ is a state vector of neurons in the second layer of a counter propagation network and $\mathbf{d}_r$ is an expected value. The basic information on training neural networks can be found in [7].

The weigths between neurons in the lattice and in the output layer are modified by (6).

$$q_{ri}^{(t)} = q_{ri}^{(t-1)} + \eta(t)(-q_{ri}^{(t-1)} + \mathbf{Y_r^{(t)}})z_i \quad (6)$$

where $\mathbf{Y_r^{(t)}}$ is evaluation of the vertex r.

Algorithm 1: Counter propagation training for data structures

- **Input:** Training set $\mathbf{T}_c$ is defined by (1).
- **Output:** Configuration of CP SOM SD network.

1. Initialize weights $\mathbf{w_i}$ to random values, for each **i**-th neuron in the lattice.
2. Initialize weights $\mathbf{u}_r$ to random values, for each r-th neuron in the output layer.
3. Repeat until the network is not in a stable stateu for each training pattern (D_l, d_l) from $\mathbf{T}_c$.
   (a) For each vertex $v \in V_{inv}(D)$
      i. Find winner neuron $\mathbf{i}^*$ according to (3).
      ii. Modify weights according the formula (4).
   (b) Compute output according to the counter propagation algorithm.
   (c) Modify weights between neurons in the lattice and in the output layer according to (6).

The space complexity of the network is $N * \log_2 K + 2 * N * K + 6 * K$, where K is the number of vertices in an acyclic data structure and N is the number of neurons in the lattice. The time complexity of the training is $l_s * N * \log_2 K$, where l_s is the number of training cycles.

6. RESULTS AND EVALUATION

From theoretical point of view, we are interested in quality of learning process of neural networks. The size of a lattice in SOM neural network is very important parameter. The differentiation of winners refers about spreading of winners in the lattice and it is dependent on its size. Confidence of right answers refers how good the network answers.

6.1. Criteria of evaluation

- **Winner differentiation WD)** [12] describes the ratio between the number of winner neurons and the number of different inputs.

$$WD = \frac{|\{j; \exists t : j = i^*(t)\}|}{N}, \quad (7)$$

If $WD < 1$, some winners are computed for different inputs. The situation is the best one if $WD = 1$.

- **Confidence of right answers.** Let p^S be the sum of all the numbers of predecessors of the vertex v_D together with the subject in the training set and p^N be the sum of all the numbers of predecessors of the vertex v_D together with the subject computed by a neural network, then the confidence of right answer is

$$d^o = \frac{p^N}{p^S}. \quad (8)$$

6.2. Analysis of queries

The student can put in the input a set of his passed subjects ($SPS = V_m$) and then he put the subject $Q = v^z$ which he would like to register for study. We analyzed the behavior of the network for the different structures of *SPS*. The set *SPS* is the set from the student point of view but in the acyclic structure of study program the subjects have their positions. The student do not put full information, it means, it is interesting to compare real information following from the study program to the information computed by network. We describe some situations:

- $|SPS| = 0, |SRS| = 0$. If the student put on input any passed subjects and he do not know any subject it means it is beginner and he will get an answer the set with the subjects without prerequisites. The situation is solved in the following query 2.
- $|SPS| = 0, SRS \geq 1$. If the student put on input any passed subject $SPS = \emptyset$ it means he would like to know if the subjects in the set *SRS* are subjects without prerequisites (in the case he should register them). The situation is solved in the following query 1.
- $|SPS| \geq 1, |SRS| = 0$. If the student put on input passed subjects only he would like to know the subjects he could register. The situation is solved in the query 2.
- $|SPS| \geq 1, |SRS| \geq 1$. It is a common situation. The student passed the set *SPS* of subjects and he would like to register the subjects from the set *SRS* (chosen set of subjects). Can he register the subject from the set *SRS*. The situation is solved in the query 1.

From the previous analysis we formulated to the following two queries:

- **Query 1:** I have passed the set of subjects *SPS*. Can I register the subject Q?
- **Query 2:** I have passed the subject *SPS*. Could I register the subject from *SRS*?

6.3. Encoding of queries

The query (input to the neural network) is prepared in the following way:

- a is the number of all subjects in the query;
- the subjects in the query are prepared as a pairs $(\mathbf{p_l}, t_l)$, where $\mathbf{p_l}$ is the code of subject and t_l is one bit (for a passed subject it contains the value 1, for not passed the value 0).

6.4. Results

We used 19 study programs from Faculty of science P. J. Šafárik university in Košice. In the test of the prepared application for CP SOM SD we did many tests for the study of a network behavior some in comparison to real values.

We construct 30 queries to each study program. Each query was built in the following way:

- The set of passed subjects $SPS = V_m$ was generated randomly from subjects in the study program V. The number of passed subjects in *SPS* was generated randomly in the interval $\langle 1, \frac{2}{3}|V|\rangle$.
- Let $V^{comp} = V - V_m$ be the set of non passed subjects. The set *SRS* represent subjects prepared to registration, $SRS \subseteq V^{comp}$ was built randomly. The number of subjects in *SRS* was computed according the rule: if $|V^{comp}| > 10$ then $|SRS| = random(7) + 3$, else $|SRS| = random(V^{comp}) + 1$.

The used real values in the training: the dimension of lattices from $N = 15 \times 15$ to 54×54: parameters α and β in interval $0.6 - 0.9$; the number of iterations 1000; the starting initial weights were chosen randomly from the interval $\langle -10; 10\rangle$; the learning rate 0.1.

Results of both quality parameters are in the Table 1 and in the Table 2. Results represent values of two defined criteria and our conclusions are the following:

- The better results were computed by networks with the bigger size of the lattice.
- The best values of parameters are: $\alpha = 0,75, \beta = 0.6$ and the size of lattice 51.

7. CONCLUSION

In the paper, we described the new developed model of the CP SOM SD neural network which could be used in the application of an academic information system. The tests of the model for acyclic data structures of study programs at university were evaluated and results give quite good starting point to the following work. The plan to the following activities is to continue in the work with an analysis of different sets *SPS* they have their special positions in the acyclic data structure and to compare it to real data.

ACKNOWLEDGEMENT

Supported by the Slovak Scientific Grant Agency VEGA, Grant No. 1/0479/12.

Table 1 Winner differentiation WD for prepared CP SOM SD

α	0,6			0,75			0,9		
β	0,6	0,75	0,9	0,6	0,75	0,9	0,6	0,75	0,9
15	0,1605	0,1539	0,1303	0,1632	0,1539	0,1553	0,1803	0,1684	0,1645
21	0,3039	0,2579	0,2526	0,2961	0,3105	0,2579	0,3211	0,3211	0,2697
27	0,4105	0,3039	0,4645	0,4342	0,3750	0,4276	0,4382	0,4461	0,3697
33	0,6197	0,5250	0,4816	0,5816	0,5816	0,5421	0,5421	0,6000	0,5026
39	0,6434	0,6447	0,6145	0,6908	0,6776	0,6408	0,6118	0,6658	0,6382
45	0,6882	0,6303	0,7605	0,6947	0,7066	0,7224	0,7461	0,6711	0,7382
51	0,7026	0,7842	0,7461	**0,8145**	0,7921	0,7171	0,7447	0,7316	0,7408
54	0,7474	0,7987	0,7382	0,7250	0,7592	0,7013	0,7329	0,7579	0,7118

Table 2 Confidence of answers in CP SOM SD

	0,6			0,75			0,9		
	0,6	0,75	0,9	0,6	0,75	0,9	0,6	0,75	0,9
15	0,2872	0,3100	0,3227	0,3786	0,3583	0,2973	0,3282	0,3335	0,3418
21	0,5114	0,5296	0,5215	0,5312	0,5644	0,5323	0,5084	0,5721	0,5416
27	0,5794	0,5865	0,7387	0,6819	0,7130	0,6946	0,6961	0,6370	0,5867
33	0,8057	0,7531	0,7765	0,8261	0,7468	0,7367	0,8123	0,8241	0,7606
39	0,8516	0,8210	0,8605	0,8425	0,8251	0,8714	0,7735	0,8506	0,8508
45	0,8107	0,8973	0,8160	0,8336	0,8457	0,8732	0,9124	0,9248	0,8694
51	0,8975	0,9244	0,9153	**0,9456**	0,9118	0,9175	0,9367	0,9248	0,9308
54	0,9288	0,8880	0,9268	0,9227	0,9401	0,9280	0,9312	0,9325	0,9292

REFERENCES

[1] ANDREJKOVÁ, G. – ORAVEC, J.: Processing Acyclic Data Structures using modified Self-Organizing Maps. In: LNCS 6692: Advances in Computational Intelligence, part II., Springer, 2011, pp. 145-152.

[2] FRASCONI, P. M. – GORI, M. – SPERDUTI, A.: A general framework of adaptive processing of data structures. IEEE-NN, 9(5), September 1998, pp. 768-786.

[3] HAGENBUCHNER, M. – SPERDUTI, A. – TSOI, A. C. : A self-organizing map for adaptive processing of structured data. IEEE Transactions on Neural Networks, 14 (3), 2003, pp. 491-505.

[4] HAGENBUCHNER, M. – TSOI, A. C.: A supervised self-organizing map for structures, Proceedings IEEE International Joint Conference on Neural Networks, 25-29 July 2004, Vol. 3, pp. 1923-1928.

[5] HAGENBUCHNER, M. – TSOI, A. C. – SPERDUTI, A.: A supervised self-organising map for structured data. In: Proc. WSOM 2001: Advances in Self-Organizing Maps. Springer, Berlin, 2001, pp. 21-28.

[6] HAMMER, B. – MICHELI, A. – SPERDUTI, A. – STRICKERT, M.: A general framework for unsupervised processing of structured data, Neurocomputing, No. 57, Elsevier, 2004, pp. 3-35.

[7] HAYKIN, S.: Neural networks: a comprehensive foundation, 2nd ed.. Prentice-Hall, New Jersey (1999).

[8] SCARSELLI, F. – GORI, M. – TSOI, A. C. – HAGENBUCHNER, M. – MONFARDINI, G.: The Graph Neural Network Model. IEEE Trans. on Neural Networks, Vol. 20, No. 1, 2009, pp. 61-80.

[9] SINČÁK, P. – ANDREJKOVÁ, G.: Neural Networks I., II., ELFA Košice, 1996 (in Slovak).

[10] SPERDUTI, A.: Tutorial on neurocomputing of structures, in Knowledge-Based Neurocomputing, I. Cloete and J. M. Zurada, Eds. Cambridge, MA: MIT Press, 2000, pp. 117-152.

[11] SPERDUTI, A.: Neural networks for adaptive processing of structured data. LNCS, 2130:5, Springer, 2001, pp. 5-12.

[12] VANČO, P. – FARKÁŠ, I.: Experimental comparison of recursive self-organizing maps for processing tree-structured data. Neurocomputing 73 (7-9), 2010, pp. 1362-1375.

Received February 22, 2012, accepted June 4, 2012

BIOGRAPHIES

Gabriela Andrejková works in Institute of Computer Science in position of associated professor. From research point of view she is interested in neural networks, their theoretical problems and applications.

Jozef Oravec is PhD. student at Faculty of Science. His thesis is oriented to neural networks for processing of structured data structures.