Математички Билтен
22 (XLVIII)
1998 (51–60)
Скопје, Македонија

ISSN 0351–336X

ILP-MODEL AND LP-MODEL OF THE NEURAL NETWORK LEARNING PROBLEM

Gjorgji Jovančevski, Dimitra Karčicka and Stevo Božinovski

Abstract

It has been shown that the problem of Neural Network Pattern Recognition Learning can be considered as an Integer Linear Programming (ILP) problem and as an Linear Programming (LP) problem.

I. Introduction

The Neural Network Pattern Recognition Learning Problem [1], [2] is stated as follows: It is needed to find the neural network memory $\overset{*}{\mathbf{W}} = \{\overset{*}{\boldsymbol{w}}_j \in R^m \mid j = 1,\dots,n\}$, so that it is possible to recognize all the vectors of a given training set $S = \{\boldsymbol{s}_j \in R^m \mid j = 1,2,\dots,n\}$ of nonnegative integer training patterns, linearly independent in pairs, such that for each pattern $\boldsymbol{s}_k$, $k \in N = \{1,2,\dots,n\}$ the inequalities

$$g_k(\overset{*}{\boldsymbol{w}}_k) > g_k(\overset{*}{\boldsymbol{w}}_j) \qquad \text{for} \qquad j \neq k\,, \tag{1}$$

are satisfied, where

$$g_k(\overset{*}{\boldsymbol{w}}_j) = \boldsymbol{s}_k^T\, \overset{*}{\boldsymbol{w}}_j + \delta_j\,, \qquad j \in N \tag{2}$$

are the activities of the appropriate neurons, caused by the patterns; δ_j, $j \in N$ are the thresholds of the neurons; $\boldsymbol{R}^m$ is the weight space.

The network learning is carried out with a training process, in which each pattern from the training set S is shown and tested if the net recognizes it or not. The training process begins with the state "tabula rasa", when the network doesn't possess any knowledge about the patterns, i.e. $\boldsymbol{w}_j = 0$, $j \in N$.

In the training process, for teaching the network to recognize the pattern $\boldsymbol{s}_k$, we use the teaching rule

$$\boldsymbol{w}_j^{(r)} = \begin{cases} \boldsymbol{w}_j^{(r-1)} + c\boldsymbol{s}_k\,, & \text{for} \quad j = k \\ \boldsymbol{w}_j^{(r-1)}\,, & \text{for} \quad j \neq k \end{cases}, \qquad j \in N$$

where c is a positive constant named learning rate [5].

II. ILP-model of the Neural Network Learning Problem

During the consideration of the neural network memory dependence on the learning rate c, there are different memory vectors obtained for different values c. For example, if $S = \{s_1, s_2, s_3\}$, where $s_1 = [1\ 0\ 2]^T$, $s_2 = [2\ 1\ 3]^T$, $s_3 = [3\ 2\ 4]^T$ for four different values of c, the appropriate memory vectors w_i, $i = 1, 2, 3$ are:

c	w_1^T	w_2^T	w_3^T
0.17	[3.57; 0.00; 7.14]	[4.42; 2.21; 6.63]	[4.59; 3.06; 6.12]
0.50	[10.50; 0.00; 21.00]	[13.00; 6.50; 19.50]	[13.50; 9.00; 18.00]
0.83	[17.43; 0.00; 34.86]	[21.58; 10.79; 32.37]	[22.41; 14.94; 29.88]
1.00	[21.00; 0.00; 42.00]	[26.00; 13.00; 39.00]	[27.00; 18.00; 36.00]

In the training process, for each of the above cases, it was necessary to make 66 trials, out of which 43 were advising trials. Anyway, the same number of advising trials were made for each pattern: 21 for the first pattern, 13 for the second and 9 for the third one. Therefore we conclude that the network memory can not be determined in a unique way, but the smallest number of network advising trials for each pattern from the training set is unique. So, in order to learn the network, a ***certain (minimal) number of advising trials (teachings) must be done for each pattern.***

This phenomenon is well known when it concerns a man. In order to recognize some pattern (of an object, phenomenon, notion ets.), the man has to have seen that pattern before, at least once. How many times the man has to see a pattern in order to remember it depends of the fact whether that pattern is similar to some other pattern that he already knows. Also, the phenomenon that it is more difficult to recognize two twins: they should be seen more times to remember them who is who than if twins were not in question.

Let $\overset{*}{p} = [\overset{*}{p}_1 \overset{*}{p}_2 \ldots \overset{*}{p}_n]^T$ be the vector of numbers of the advising trials for the appropriate patterns $s_1, s_2, \ldots, s_n$ after the training process is finished.($\overset{*}{p}$ is a ***teaching vector*** from the n-dimensional Euclidean space; R^n is called a ***teching space*** or ***teacher's space.***) Then, the memory network vectors are

$$\overset{*}{w}_j = c\ \overset{*}{p}_j\ s_j\,, \qquad j \in N \tag{3}$$

and $\overset{*}{p}_j \geq 1$, $j \in N$ are positive integers.

According to (3), the conditions (1) and (2) for the vector recognition s_k, can be given by the teaching vector p, as follows:

$$g_{kk}(p) > g_{kj}(p) \quad \text{for all} \quad j \neq k \tag{4}$$

$$g_{kj}(p) = cs_k^T s_j e_j^T p + \delta_j \quad \text{for all} \quad j \in N \tag{5}$$

where $\boldsymbol{e}_j,\ j \in N$ are the unit vectors from R^n.

By denoting

$$\boldsymbol{h}_{kj}^T = [0 \dots 0\ h_{kk}\ 0 \cdots 0 - h_{kj} 0 \dots 0]$$

where $h_{kj} = \boldsymbol{s}_k^T \boldsymbol{s}_j \geq 0,\ h_{kk} = \boldsymbol{s}_k^T \boldsymbol{s}_k > 0$ and under the assumption that the neurons have the same inner potential $\delta = \delta_j,\ j \in N$, condition (4) can be expressed as a scalar product of $\boldsymbol{h}_{kj}$ and $\boldsymbol{p}$,

$$\boldsymbol{h}_{kj}^T \boldsymbol{p} > 0\,. \tag{6}$$

For each integer positive vector $\boldsymbol{x} = \boldsymbol{p}$, the scalar product $\boldsymbol{h}_{kj}^T \boldsymbol{x}$ is integer. Then, condition $\boldsymbol{h}_{kj}^T \boldsymbol{x} > 0$ means that $\boldsymbol{h}_{kj}^T \boldsymbol{x}$ is an integer positive number. So, for $\boldsymbol{x} \geq \boldsymbol{e} = [1 \dots 1]^T$ and $\boldsymbol{x}$-integer, we get $\boldsymbol{h}_{kj}^T \boldsymbol{p} \geq 1$.

This means that the teaching vectors are elements of the set

$$D = \{\boldsymbol{x} \in R^n \mid \boldsymbol{h}_{kj}^T \boldsymbol{x} \geq 1\,, \quad \boldsymbol{x} \geq \boldsymbol{e}, k, j \in N, \quad k \neq j, \ \ \boldsymbol{x} - \text{integer}\}\,.$$

The choice of the learning rule guarantees the existence of a final teaching vector $\overset{*}{\boldsymbol{p}}$ (with final positive integer components) for recognizing each pattern from S.

Let $\boldsymbol{p} = [p_i]_{nx1}$ be a known (in some way) teaching vector i.e. $\boldsymbol{p} \in D$. Then, further network training is not necessary, because the neural network memory W, which recognizes each pattern from S, is defined. But, such vectors are $\gamma \boldsymbol{p}$ for each integer $\lambda > 0$. Therefore, a most rational choice of the teaching vector will be $\overset{*}{\boldsymbol{p}} = [\overset{*}{p}]_{nx1}$, which is got with the smallest number of network teachings for the patterns from the training set S. This means that sum of the components of $\overset{*}{\boldsymbol{p}}$,

$$\overset{*}{z} = \overset{*}{p}_1 + \cdots + \overset{*}{p}_n \quad \text{has to be minimal.}$$

So,

$$\overset{*}{z} = \boldsymbol{e}^T \overset{*}{\boldsymbol{p}} \leq \boldsymbol{e}^T \boldsymbol{p} \quad \text{for} \quad \forall \boldsymbol{p} \in D\,.$$

Then, $\overset{*}{z} = \boldsymbol{e}^T \overset{*}{\boldsymbol{p}}$ is the minimal value of the linear funcional $z = \boldsymbol{e}^T \boldsymbol{x}$ on D.

So, we get the formulation of the neural network learning problem as solvable integer Linear Programming Problem (ILPNN-problem):

$$\min\{z = \boldsymbol{e}^T \boldsymbol{x} \mid \boldsymbol{h}_{kj}^T \boldsymbol{x} \geq 1, \quad k, j \in N, \quad k \neq j, \quad \boldsymbol{x} \geq \boldsymbol{e}, \quad \boldsymbol{x} - \text{integer}\} \tag{7}$$

This problem can be written in the following compact form

$$\min\{z = \boldsymbol{e}^T \boldsymbol{x} \mid \begin{bmatrix} A \\ I \end{bmatrix} \boldsymbol{x} \geq \begin{bmatrix} f \\ \boldsymbol{e} \end{bmatrix}, \quad \boldsymbol{x} - \text{integer}\}$$

where I is the identity matrix of the n-th order; $\boldsymbol{e}$ is the n-vector with components all equal to 1; $\boldsymbol{f}$ is the $n(n-1)$-vector with components all equal to 1, and A is $(n-1)nxn$ matrix of the following structure: $A = [A_1\ A_2 \dots A_k \dots A_n]^T$. The block $A_k = [a_{ij}^{(k)}]_{(n-1)xn}$ has the shape

$$A_k = \begin{bmatrix} -h_{k1} & 0 & \cdots & 0 & h_{kk} & 0 & \dots & 0 \\ 0 & -h_{k2} & \cdots & 0 & h_{kk} & 0 & \dots & 0 \\ \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdots & \cdot \\ 0 & 0 & \cdots & -h_{k,k-1} & h_{kk} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & h_{kk} & -h_{k,k+1} & \cdots & 0 \\ \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdots & \cdot \\ 0 & 0 & \cdots & 0 & h_{kk} & 0 & \cdots & -h_{kn} \end{bmatrix} \quad (8)$$

III. LP-model of the Neural Network Learning Problem

For an optimal program $\hat{\boldsymbol{x}} = [\hat{x}_j]$ and the optimal value $\hat{z} = \boldsymbol{e}^T\hat{\boldsymbol{x}}$ of (7), the following conditions are satisfied:

$\boldsymbol{e}^T \left[\frac{1}{\hat{z}}\hat{\boldsymbol{x}}\right] = 1$ and $\frac{1}{\hat{z}} \geq \frac{1}{\boldsymbol{e}^T\boldsymbol{x}}$, for each program $\boldsymbol{x}$.

Let $y_{n+1} = \frac{1}{\hat{z}}$, $y_j = \frac{\hat{\boldsymbol{x}}_j}{\hat{z}}$, $j = 1, \dots, n$, and $\boldsymbol{y} = [y_1, \dots, y_n]^T$.

Then directly from ILPNN-problem we obtain LP-problem [3].

$$\begin{aligned} &\text{Maximize} && w = y_{n+1} \\ &\text{subject to} && -A_i\boldsymbol{y} + y_{n+1}h_i\boldsymbol{e}^{(n-1)} \leq \boldsymbol{o}^{(n-1)}, \quad i = 1, \dots, n \\ &&& -\boldsymbol{y} + y_{n+1}\boldsymbol{e}^{(n)} \leq \boldsymbol{o}^{(n)} \\ &&& (\boldsymbol{e}^{(n)})^T\boldsymbol{y} = 1 \\ &&& \boldsymbol{y} \geq \boldsymbol{o}^{(n)}, \quad y_{n+1} \geq 0. \end{aligned}$$

where,

$$A_i = \begin{bmatrix} -a_{11} & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdots & \cdot \\ 0 & 0 & \cdots & -a_{i-1,i} & 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 1 & -a_{i+1,i} & \cdots & 0 \\ \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & -a_{ni} \end{bmatrix}, \qquad h_i = \frac{1}{h_{ii}}, \quad a_{si} = \frac{h_{si}}{h_{ii}}, \quad s \neq i$$

$\boldsymbol{e}^{(s)}$ is the s-vector with components all equal to 1; $\boldsymbol{o}^{(s)}$ denotes the zero vector of size s; $\boldsymbol{y} = [y_j]_{nx1}$ is the vector of unknowns as y_{n+1}.

We notice that y_j, $j = 1, \ldots, n$ can be interpreted as frequencies of the optimal pattern appearances in the learning process, ($0 \leq y_j \leq 1$ and $(\boldsymbol{e}^{(n)})^T \boldsymbol{y} = 1$).

We observe that the number of constraints in LP is much larger than the number of variables. Therefore, it is useful to state and to consider the dual to LP i.e. DP:

$$\begin{aligned}
&\text{Minimize} && \zeta = t \\
&\text{subject to} && \sum_{i=1}^{n} (-A_i)^T \boldsymbol{u}_i - \boldsymbol{v} + t\boldsymbol{e}^{(n)} \geq \boldsymbol{o}^{(n)} \\
& && \sum_{i=1}^{n} h_i (\boldsymbol{e}^{(n-1)})^T \boldsymbol{u}_i + (\boldsymbol{e}^{(n)})^T \boldsymbol{v} = 1 \\
& && \boldsymbol{v} \geq \boldsymbol{o}^{(n)}, \quad \boldsymbol{u}_i \geq \boldsymbol{o}^{(n-1)}, \quad i = 1, \ldots, n;
\end{aligned}$$

there is no restriction on the sign of the dual variable t.

Let t be substituted by two nonnegative variables $t^+ = \max\{0, t\}$, and $t^- = \max\{0, -t\}$, $t = t^+ - t^-$, and let q be the slack vector, the components of which express inequality constraints in DP,

$$\boldsymbol{q} = \sum_{i=1}^{n} (-A_i)^T \boldsymbol{u}_i - \boldsymbol{v} + t^+ \boldsymbol{e}^{(n)} - t^- \boldsymbol{e}^{(n)} \qquad (\geq \boldsymbol{o}^{(n)}),$$

Then, DP becomes an LP-problem in standard form, with equality constraints and extended matrix

$$[\hat{A}, \hat{d}] = \begin{bmatrix} -A_1^T & \cdots & -A_n^T & -\boldsymbol{I}^{(n)} & \boldsymbol{e}^{(n)} & -\boldsymbol{e}^{(n)} & -\boldsymbol{I}^{(n)} & \boldsymbol{o}^{(n)} \\ h_1(\boldsymbol{e}^{(n-1)})^T & \cdots & h_n(\boldsymbol{e}^{(n-1)})^T & (\boldsymbol{e}^{(n)})^T & 0 & 0 & (\boldsymbol{o}^{(n)})^T & 1 \end{bmatrix}$$

where $\boldsymbol{I}^{(n)}$ denotes the identity nxn matrix; for $i = 1, \ldots, n$, the blocks in i-th column of $\hat{A}$, $-A_i^T$ and $h_i(\boldsymbol{e}^{(n-1)})^T$ represent the submatrix

$$\begin{vmatrix}
a_{1i} & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\
\cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdots & \cdot \\
0 & 0 & \cdots & a_{i-1,i} & 0 & 0 & \cdots & 0 \\
-1 & -1 & \cdots & -1 & -1 & -1 & \cdots & -1 \\
0 & 0 & \cdots & 0 & a_{i-1,i} & 0 & \cdots & 0 \\
\cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdots & \cdot \\
0 & 0 & \cdots & 0 & 0 & 0 & \cdots & a_{ni} \\
h_i & h_i & \cdots & h_i & h_i & h_i & \cdots & h_i
\end{vmatrix}$$

The known methods seem to be inefficient when applied to (ILPNN), (LP), (DP), since the good properties, especially the matrix sparsity, will be destroyed.

The application of the known ILP algorithms for solving the ILPNN-problem is associated with the difficulties which result out from the limitation of the computer's memory. For example, we can apply Gomory's Discrete Algorithm, but this algorithm may be efficient for a small number of patterns n, only.

IV. Ilustration

Let $n = 4$ and the training set S consist of the following patterns $s_j \in R^{63}$, $j = 1,2,3,4$:

$$s_1 = [000000001111100100000010000001000000100000010000001000000000000]^T$$
$$s_2 = [000100001111100100000010000001000000100000010000001000000000000]^T$$
$$s_3 = [000000001111100100000010000001111100100000010000001111100000000]^T$$
$$s_4 = [000000001111100100010010001001000100100010010001001000100000000]^T$$

After 15 iterations, an optimal program $\overset{*}{p} = [13\ 12\ 8\ 9]^T$ was obtained, with optimal value of the objective function $z = 42$. The strategy of searching the optimal program is:

		x_1	x_2	x_3	x_4		z
1	$x = [$	0	0	0	0	$]^T =$	0
2	$x = [$	0	0	0	1	$]^T =$	1
3	$x = [$	0	0	1	1	$]^T =$	2
4	$x = [$	0	1	1	1	$]^T =$	3
5	$x = [$	2	1	1	1	$]^T =$	5
6	$x = [$	2	1	1	2	$]^T =$	6
7	$x = [$	2	1	2	2	$]^T =$	7
8	$x = [$	3	1	2	2	$]^T =$	8
9	$x = [$	3	3	2	2	$]^T =$	10
10	$x = [$	4	3	2	2	$]^T =$	11
11	$x = [$	4	3	2	3	$]^T =$	12
12	$x = [$	4	3	3	3	$]^T =$	13
13	$x = [$	13	12	3	3	$]^T =$	31
14	$x = [$	13	12	3	9	$]^T =$	37
15	$\overset{*}{p} = [$	13	12	8	9	$]^T =$	42

The simplicity of the essential constraints of the ILPNN-problem stimulated us to search for a new ILP algorithm. We investigate a new algorithm, which is not affected by the limited computer's performances. It is confirmed that finding the optimal solution is very fast.

OBJECT PATTERNS – LETTERS FROM THE ALPHABET IN A BINARY FORM

A	[000000000010000010100010001001000100111110010001001000100000000]^T
B	[000000001111000100010010001001111000100010010001001111000000000]^T
C	[000000000111000100010010000001000000100000010001000111000000000]^T
D	[000000001111000100010010001001000100100010010001001111000000000]^T
E	[000000001111100100000010000001111100100000010000001111100000000]^T
F	[000000001111100100000010000001111100100000010000001000000000000]^T
G	[000000000111000100010010000001011100100010010001000111000000000]^T
H	[000000001000100100010010001001111100100010010001001000100000000]^T
I	[000000000111000001000000100000010000001000000100000111000000000]^T
J	[000000000000100000010000001000000100000010010001000111000000000]^T
K	[000000001000100100100010100001100000101000010010001000100000000]^T
L	[000000001000000100000010000001000000100000010000001111100000000]^T
M	[000000010000011100011101010110010011000001100000110000010000000]^T
N	[000000001000100100010011001001010100100110010001001000100000000]^T
O	[000000000111000100010010001001000100100010010001000111000000000]^T
P	[000000001111000100010010001001111000100000010000001000000000000]^T
Q	[000000000111000100010010001001000100100010010011000111100000001]^T
R	[000000001111000100010010001001111000101000010010001000100000000]^T
S	[000000000111000100010010000000111000000010010001000111000000000]^T
T	[000000001111100001000000100000010000001000000100000010000000000]^T
U	[000000001000100100010010001001000100100010010001000111000000000]^T
V	[000000001000100100010010001001000100100010001010000010000000000]^T
W	[000000010000011000001100000110000011000001100100101101100000000]^T
X	[000000001000100100010001010000010000010100010001001000100000000]^T
Y	[000000001000100100010001010000010000001000000100000010000000000]^T
Z	[000000001111100000010000010000010000010000010000001111100000000]^T

SEARCHING THE OPTIMAL VECTOR

$$
\begin{array}{l|cccccccccccccccccccccccccc|l}
 & A & B & C & D & E & F & G & H & I & J & K & L & M & N & O & P & Q & R & S & T & U & V & W & X & Y & Z & \\
p=[& 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 &]^T \\
p=[& 0 & 0 & 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 &]^T \\
p=[& 0 & 0 & 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 &]^T \\
p=[& 0 & 0 & 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 &]^T \\
p=[& 0 & 8 & 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 &]^T \\
p=[& 0 & 8 & 0 & 9 & 0 & 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 &]^T \\
p=[& 0 & 8 & 11 & 9 & 0 & 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 &]^T \\
p=[& 0 & 8 & 11 & 9 & 0 & 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 9 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 &]^T \\
p=[& 0 & 8 & 11 & 9 & 0 & 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 9 & 9 & 8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 &]^T \\
p=[& 0 & 8 & 11 & 9 & 0 & 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 9 & 9 & 8 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 &]^T \\
p=[& 0 & 8 & 11 & 9 & 7 & 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 9 & 9 & 8 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 &]^T \\
p=[& 0 & 8 & 11 & 9 & 7 & 8 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 9 & 9 & 8 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 &]^T \\
p=[& 0 & 8 & 11 & 9 & 7 & 8 & 9 & 7 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 9 & 9 & 8 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 &]^T \\
p=[& 0 & 8 & 11 & 9 & 7 & 8 & 9 & 7 & 0 & 0 & 0 & 0 & 0 & 7 & 10 & 9 & 9 & 8 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 &]^T \\
p=[& 0 & 8 & 11 & 9 & 7 & 8 & 9 & 7 & 0 & 0 & 0 & 0 & 0 & 7 & 10 & 9 & 9 & 8 & 9 & 0 & 9 & 0 & 0 & 0 & 0 & 0 &]^T \\
p=[& 0 & 8 & 11 & 9 & 7 & 8 & 9 & 7 & 0 & 10 & 0 & 0 & 0 & 7 & 10 & 9 & 9 & 8 & 9 & 0 & 9 & 0 & 0 & 0 & 0 & 0 &]^T \\
p=[& 0 & 8 & 11 & 9 & 7 & 8 & 9 & 7 & 0 & 10 & 0 & 9 & 0 & 7 & 10 & 9 & 9 & 8 & 9 & 0 & 9 & 0 & 0 & 0 & 0 & 0 &]^T \\
p=[& 0 & 8 & 11 & 9 & 7 & 8 & 9 & 7 & 0 & 10 & 0 & 9 & 0 & 7 & 10 & 9 & 9 & 8 & 9 & 0 & 9 & 8 & 0 & 0 & 0 & 0 &]^T \\
p=[& 6 & 8 & 11 & 9 & 7 & 8 & 9 & 7 & 0 & 10 & 0 & 9 & 0 & 7 & 10 & 9 & 9 & 8 & 9 & 0 & 9 & 8 & 0 & 0 & 0 & 0 &]^T \\
p=[& 6 & 8 & 11 & 9 & 7 & 8 & 9 & 7 & 0 & 10 & 7 & 9 & 0 & 7 & 10 & 9 & 9 & 8 & 9 & 0 & 9 & 8 & 0 & 0 & 0 & 0 &]^T \\
p=[& 6 & 8 & 11 & 9 & 7 & 8 & 9 & 7 & 0 & 10 & 7 & 9 & 0 & 7 & 10 & 9 & 9 & 8 & 9 & 0 & 9 & 8 & 0 & 6 & 0 & 0 &]^T \\
p=[& 6 & 8 & 11 & 9 & 7 & 8 & 9 & 7 & 0 & 10 & 7 & 9 & 0 & 7 & 10 & 9 & 9 & 8 & 9 & 0 & 9 & 8 & 0 & 6 & 0 & 7 &]^T \\
p=[& 6 & 8 & 11 & 9 & 7 & 8 & 9 & 7 & 0 & 10 & 7 & 9 & 0 & 7 & 10 & 9 & 9 & 8 & 9 & 0 & 9 & 8 & 0 & 6 & 5 & 7 &]^T \\
p=[& 6 & 8 & 11 & 9 & 7 & 8 & 9 & 7 & 0 & 10 & 7 & 9 & 0 & 7 & 10 & 9 & 9 & 8 & 9 & 5 & 9 & 8 & 0 & 6 & 5 & 7 &]^T \\
p=[& 6 & 8 & 11 & 9 & 7 & 8 & 9 & 7 & 0 & 10 & 7 & 9 & 0 & 7 & 10 & 9 & 9 & 8 & 9 & 6 & 9 & 8 & 0 & 6 & 5 & 7 &]^T \\
p=[& 6 & 8 & 11 & 9 & 7 & 8 & 9 & 7 & 7 & 10 & 7 & 9 & 0 & 7 & 10 & 9 & 9 & 8 & 9 & 6 & 9 & 8 & 0 & 6 & 5 & 7 &]^T \\
p=[& 6 & 8 & 11 & 9 & 7 & 8 & 9 & 7 & 7 & 10 & 7 & 9 & 2 & 7 & 10 & 9 & 9 & 8 & 9 & 6 & 9 & 8 & 0 & 6 & 5 & 7 &]^T \\
\overset{*}{p}=[& 6 & 8 & 11 & 9 & 7 & 8 & 9 & 7 & 7 & 10 & 7 & 9 & 2 & 7 & 10 & 9 & 9 & 8 & 9 & 6 & 9 & 8 & 3 & 6 & 5 & 7 &]^T
\end{array}
$$

References

[1] Božinovski S.: *CAA Neural Modules And Modeling The Pattern Classification Reaching Task: Teaching Grammars, Teaching Model and Similarity Model*, in NEURAL NETWORKS Concepts, Applications, and Implemenatations, Volume IV, P. Antognetti and V. Milutinović, Editors, 1-23, Prentice Hall, 1991.

[2] Jovančevski Gj, Karčicka D., Božinovski S.: *Neural network training as a special integer linear programming problem*, LIRA'95, Novi Sad, 26-30 September, FR Yugoslavia.

[3] Karčicka D, Jovančevski Gj.: *LP-Model of neural network training problem in the teaching space*, LIRA'95, Novi Sad 26-30 September, FR Yugoslavia.

[4] Karmanov V.G.: *Mathematical programming*, Nauka, Moscow, 1980.

[5] Simpson P.: *Artificial neural systems: Foundations, Paradigms, Applications, and Implementations*, Pergamon Press, San Diego, 1990.

[6] Schrijver A.: *Theory of linear and integer programming*, John Wiley & Sons, 1986.

[7] Hush D, and Horne B.: *Progress in supervised neural networks*, IEEE Signal processing magazine, 8-39, January, 1993.

ИЛП–МОДЕЛ И ЛП–МОДЕЛ НА ПРОБЛЕМОТ НА ОБУЧУВАЊЕ НЕВРОНСКА МРЕЖА

Ѓорѓи Јованчевски, Димитра Карчицка и Стево Божиновски

Р е з и м е

При обучување на невронска мрежа за препознавање на ликови е констатирано дека за едно обучувачко множество $S = \{s_j \mid j = 1, \ldots, n\}$ е потребен одреден (минимален) број на обучувања (учења) p_i обиди за секој лик s_i. За наоѓање на таков вектор на учење $\overset{*}{p} = [\overset{*}{p}_1 \overset{*}{p}_2 \ldots \overset{*}{p}_n]^T$ може да се дефинира ИЛП-задача

$$\min\{z = e^T x \mid h_{kj}^T x \geq 1\,, \quad k, j \in N, \;\; k \neq j, \;\; x \geq e, \;\; x\,\text{–integer}\}$$

ЛП-задача и ДП-задача. Решавањето на ИЛП-задачата со дискретниот алгоритам на Gomory дава добри резултати само за мало множество ликови, и затоа е развиен ИЛП алгоритам кој е многу поефикасен.

Faculty of Natural Sciences and Mathematics,
Institute of Informatics
St. Cyril and Methodius University,
P.Box 162,
91000 Skopje,
Macedonia

gjorgji@robig.pmf.ukim.edu.mk

dimitra@robig.pmf.ukim.edu.mk