Buletinul Ştiinţific al Universităţii "Politehnica" din Timişoara

Seria ELECTRONICĂ şi TELECOMUNICAŢII
TRANSACTIONS on ELECTRONICS and COMMUNICATIONS

Tom 49(63), Fascicola 1, 2004

A Syntactical Self-Organizing Map with Levensthein Metrics and Its Application for Automatic Translation

Victor-Emil Neagoe[1]

Abstract –We present an original approach to the design and test of a *Syntactical Self-Organizing Map (SSOM)*. The proposed SSOM is based on Levensthein metrics and it uses a symbolic representation for both the inputs and also the net weights. We have chosen a ring architecture of the map. The software implementation of the experimental model is designed for automatic Romanian-English translation of 1009 words and expressions. We have evaluated the error correction capabilities of the system.

Keywords: syntactical self-organizing map, Levensthein distance, automatic translation.

I. INTRODUCTION

There are many automatic translation systems using very sophisticated techniques, most of them being non-neural. They take into account both the dictionary of correspondence between the words of the two languages and also the syntax rules. The design of such a system is laborious and usually it is specific for the two languages. On the other side, in pattern recognition, the Self-Organizing Map (SOM) (often called Kohonen network [1], [2]) performs a high quality classification, assigning the similar input vectors to the same neuron or to neighbor neurons. Thus, this network transforms the relation of *similarity* between input vectors into a relation of *neighborhoodness* of the neurons. The map uses the competition principle, by evaluating the distance between the input vector and the weight vector corresponding to each neuron.

We further present an original approach for automatic translation of words and expressions using a Syntactical Self-Organizing Map (SSOM), tolerant to input (typing) errors.

II. A SYNTACTICAL SELF-ORGANIZING MAP BASED ON LEVENSTHEIN METRICS

Characteristics. The proposed **Syntactical Self-Organizing Map (SSOM)** has significant differences by comparison to the conventional SOM. It has the following **main characteristics**:

• Both the inputs and also the weights of the SSOM are represented into a symbolic form, but not in a numerical one (like for SOM);

• By introducing SSOM instead of SOM, we eliminate the condition that the two representations used in the competition phase to have the same length. Instead of evaluating the Euclidean distance between two real vectors, belonging to the same space (for the case of conventional SOM), we evaluate the *weighted Levensthein distance* between two rows of symbols with different lengths, for the SSOM. For computing the above distance, we have used the Wagner-Fisher algorithm . The application of Levensthein distance is perfectly adequate for our task, where we compare words (expressions) of different lengths.

• The SSOM uses the *competition principle* (like SOM). One computes the Levenshein distances between the input row of symbols (for example, corresponding to an input word to be translated) and all the rows of weights corresponding to the network neurons. The winner is the neuron that minimizes the above distances.

Levensthein metrics. We further define the *weighted Levensthein metrics* ([3], [5], [6]) in order to compute the distance between two rows of symbols of different lengths. Let α and β be two rows of symbols of lengths "m" and "n " defined as

$$\alpha = a_1 a_2 \ldots a_m$$

$$\beta = b_1 b_2 \ldots b_n$$

[1] Faculty of Electronics, Telecommunications, and Information Technology, Polytechnic University of Bucharest, Spl. Independentei Nr. 313, Bucharest, Romania, e-mail: victor@ifia.pub.ro

where a_i, b_j are symbols belonging to the same alphabet. We define the transformation

$$M : \alpha \to \beta$$

as being the set of ordered pairs (i, j), where $1 \le i \le m, 1 \le j \le n$. We denote

$$I = \{i \mid (i, j) \in M\}$$

$$J = \{j \mid (i, j) \in M\}$$

One can prove that the M transformation satisfies the following interpretations:

(1) If $a_i \neq b_j$, for $(i, j) \in M$, then b_j is substituted by a_i;

(2) If $j \notin J$, then b_j is inserted;

(3) If $i \notin I$, then a_i is eliminated.

Let M be the set of all the transformations from α to β. Then, the minimum cost of a transformation from α to β, denoted by $D(\alpha, \beta)$ is defined as follows:

$$D(\alpha, \beta) = \min_{M \in M} \left[\sum_{(i,j) \in M} p(i, j) + (n - |J|) * q + (m - |I|) * r \right]$$

where $|I|, |J|$ are the cardinals of I, respectively of J, q is the insertion cost, r is the elimination cost, and

$$p(i, j) = \begin{cases} 0, a_i = b_j \\ p, a_i \neq b_j \end{cases}$$

is the cost of substitution of symbol a_i by b_j.

The above-defined distance is called *weighted Levensthein distance if the following two conditions are fulfilled for any pair* $(i_1, j_1), (i_2, j_2)$:

(1) $i_1 = i_2 \Leftrightarrow j_1 = j_2$

(2) $i_1 < i_2 \Leftrightarrow j_1 < j_2$

Wagner-Fisher algorithm. In order to iteratively compute the Levensthein distance, we have used the Wagner-Fisher algorithm (given in [3], [5], [6]), having the following steps:

Step 1. D (0,0) = 0;

Step 2. For i from 1 to n, one computes

D (i, 0)=D (i-1, 0)+ $c_1(a_i)$;

Step 3. For j from 1 to m, one computes

D (0,j)=D (0,j-1)+ $c_2(b_j)$;

Step 4. For i from 1 to n and j from 1 to m, one computes:

$$A_1 = D(i-1, j-1) + c(a_i, b_j);$$

$$A_2 = D(i-1, j) + c_1(a_i);$$

$$A_3 = D(i, j-1) + c_2(b_j);$$

$$D(i, j) = \min(A_1, A_2, A_3);$$

where $c_1(a_i), c_2(b_j)$, and $c(a_i, b_j)$ are the costs of insertion, elimination, and respectively substitution.

Step 5. The distance between the two words is D (n, m).

SSOM Training. Training of the SSOM (refinement of the weights) is performed by analogy to the procedure of conventional SOM, in a neighborhood of the winner neuron, using the Levensthein distance between the input row and the row of weights characterizing the winner neuron (or one of its neighbors) . One tends to change the weight row in order to minimize the Levensthein distance. To solve this task, one uses the following operations: insertion, deletion and/or substitution.

The training algorithm for SSOM is the following:

Step 1.

• *Initialize the weights of the SSOM. For the case of automatic translation application, one can choose for initialization the symbol corresponding to the "space" (the 27-th letter of the chosen alphabet, see the experimental results).*

Step 2.

• *Apply one by one the input rows of symbols (words or expressions) belonging to the training set.*

• *For each of them, compute the winner neuron, by minimizing the Levensthein distance between the input word and all the weight rows corresponding to the SSOM neurons.*

• *Make identical the weight row of the winner neuron with that of the input word (corresponding row).*

• *Refine the weight rows of the neurons belonging to the neighborhood of the winner by performing the elementary operations of substitution, insertion, and deletion, in order to reduce their Levensthein distances to the input word (but not to make them zero). The reduction of the Levensthein distance is a function of the neuron position regarding the winner; this reduction increases when the Euclidean distance between the corresponding neuron and the winner neuron decreases.*

Step 3.

• *Compute the classification error as a sum of all the minimum Levenshein distances of the words (expressions) belonging to the training lot. Such a distance is the minimum of the distances between the corresponding input word and the weight rows of the SSOM neurons.*

Step 4.

•*Test the stop condition (if the classification error is zero)*

SSOM Test. The test (operational) phase consists of computing the Levensthein distance between the input word and all the weight rows of the SSOM. The minimum distance leads to the winner neuron; its label corresponds to the class label.

III. EXPERIMENTAL RESULTS

We further present the experimental results of the software implementation of the model, corresponding to a Romanian-English translation application, using a

set of 1009 Romanian words and expressions. One chooses a circular architecture of the SSOM (Fig. 1). Denoting by "*n* " the number of inputs (representing the maximum number of symbols belonging to an input row) and by "*M*" he number of outputs (neurons), the software experiment corresponds to: *n =20* and *M= 1050*. The weight matrix is *w(i,j), i= 1, 2, ..., n; j=1, 2, ...,M.*

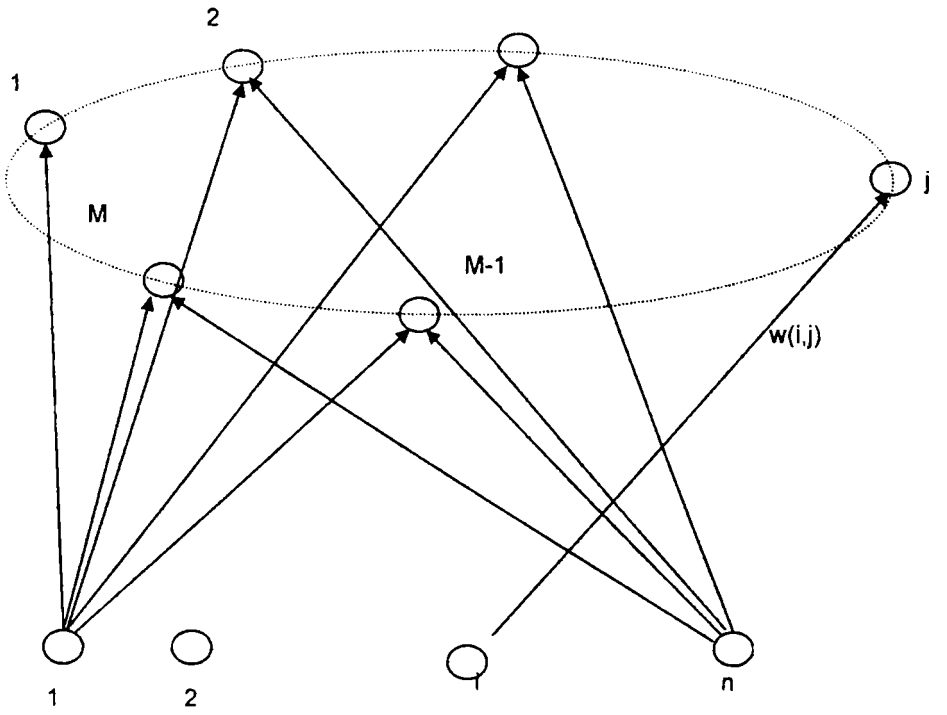

Fig. 1. The circular architecture of the Syntactical Self-Organizing Map (SSOM)

0	5	3	2	2	3	4	5	7	6	7	8	7	6	8	9	1	3	1	4	6	4	1	2	5	1	3
5	0	2	3	4	2	1	1	4	2	3	4	2	1	5	5	6	3	4	2	2	1	5	3	2	4	1
3	2	0	1	2	1	2	3	5	4	5	6	4	3	6	7	4	2	2	2	4	1	3	1	3	2	1
2	3	1	0	1	1	2	3	5	4	5	6	5	4	6	7	3	1	1	2	4	2	2	1	3	2	2
2	4	2	1	0	2	3	4	5	5	6	7	6	5	6	7	2	1	1	2	4	3	1	2	3	2	3
3	2	1	1	2	0	1	2	4	3	4	5	4	3	5	6	4	1	2	1	3	1	3	2	2	3	2
4	1	2	2	3	1	0	1	3	2	3	4	3	2	4	5	6	2	3	1	2	1	4	3	1	4	2
5	1	3	3	4	2	1	0	2	1	2	3	2	1	3	4	6	3	4	1	1	2	5	4	1	5	2
7	4	5	5	5	4	3	2	0	1	1	1	2	2	1	2	7	4	5	3	1	4	6	6	2	7	3
6	2	4	4	5	3	2	1	1	0	1	2	1	1	2	3	7	4	5	3	1	3	6	5	2	6	2
7	3	5	5	6	4	3	2	1	1	0	1	1	2	1	2	8	5	6	4	2	4	7	6	3	7	2
8	4	6	6	7	5	4	3	1	2	1	0	2	3	1	1	9	6	7	5	3	5	8	7	4	8	2
7	2	4	5	6	4	3	2	2	1	1	2	0	1	2	3	8	5	6	4	2	3	7	5	3	6	1
6	1	3	4	5	3	2	1	2	1	2	3	1	0	3	4	7	4	5	3	2	2	6	4	2	5	1
9	5	6	6	6	5	4	3	1	2	1	1	2	3	0	1	8	5	7	4	2	5	7	7	3	8	3
9	5	7	7	7	6	5	4	2	3	2	1	3	4	1	0	9	6	8	5	3	6	8	8	4	9	3
1	6	4	3	2	4	6	6	7	7	8	9	8	7	8	9	0	3	2	4	6	5	1	3	5	2	3
3	3	2	1	1	1	2	3	4	4	5	6	5	4	5	6	3	0	2	1	3	2	2	2	2	3	3
1	4	2	1	1	2	3	4	5	5	6	7	6	5	7	8	2	2	0	3	5	3	1	1	4	1	2
4	2	2	2	2	1	1	1	3	3	4	5	4	3	4	5	4	1	3	0	2	2	3	3	1	4	3
6	2	4	4	4	3	2	1	1	1	2	3	2	2	2	3	6	3	5	2	0	3	5	5	1	5	3
4	1	1	2	3	1	1	2	4	3	4	5	3	2	5	6	5	2	3	2	3	0	4	2	2	3	1
1	5	3	2	1	3	4	5	6	6	7	8	7	6	7	8	1	2	1	3	5	4	0	2	4	2	3
2	3	1	1	2	2	3	4	6	5	6	7	5	4	7	8	3	2	1	3	5	2	2	0	4	1	1
5	2	3	3	3	2	1	1	2	2	3	4	3	2	3	4	5	2	4	1	1	2	4	4	0	5	3
1	4	2	2	2	3	4	5	7	6	7	8	6	5	8	9	2	3	1	4	5	3	2	1	5	0	2
3	1	1	2	3	2	2	2	3	2	2	2	1	1	3	3	3	3	2	3	3	1	3	1	3	2	0

Fig. 2. The "27 x 27" matrix of substitution costs

The input alphabet has 27 symbols, namely 26 letters and the "space" symbol (for expression translation):

A={ 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 }.

For using the Levensthein metrics, we have chosen the substitution costs, insertion costs, and deletion costs. The substitution costs are defined as a function of the distance between the corresponding letters on the keyboard, in order to correct the key stroke (typing) errors (for example, c (a, b)=5, c (a, c) =3, c (a, d) =2). Taking into account the above principle, we have defined the corresponding 27 x 27 matrix of substitution costs. (Fig. 2). The insertion, respectively the deletion costs are taken all equal to 2.

For training, we used a variable size neighborhood, where the radius of neighborhood decreases with the epoch index, according to the following sequence:

{ 256 , 256 , 256 , 128 , 128 , 128 , 32 , 32 , 32 , 8 , 8 , 8 , 2 , 2 , 2 }.

To perform an automatic translation, we associate a table to the output labeled neurons giving the correspondence between the two languages: Romanian and English. At the same time, taking into account the SSOM design, one can correct some input (typing) errors. In Table 1, we give an example. [4]

Table 1. Example of automatic translation and error correction:

Original Word (Ro)	Erroneous Input (Ro)	Output (En)	Error Correction
abil	abi	skilful	yes
abur	abuv	steam	yes
lemn	lenm	wood	yes
admite	dmit	bribe	no

IV. CONCLUDING REMARKS

[8]

1. We propose a Syntactical Self-Organizing Map (SSOM) for automatic translation that can receive input words or expressions represented by rows of symbols of different lengths, by eliminating the necessity of normalizing the length of input rows.

2. The syntactical way of representation of the input words and expressions is *naturally* adequate to the automatic translation. One uses the specific weighted Levensthein distance, which is computed by applying the Wagner-Fisher algorithm. For the present application, any substitution cost given in the matrix of Fig. 1 is defined as a function of the Euclidean distance between the corresponding letters on the keyboard, in order to correct the input typing errors.

3. An important advantage of the model is that its design does not depend on the specific two languages, characterizing the translation. Consequently, it can be trained and retrained by the user for any pair of languages.

ACKNOWLEDGMENTS

The author wishes to thank his former skilful students Oana Cula and Bogdan Georgescu, for their contribution to the implementation of the software corresponding to the presented model.

REFERENCES

[1] T. Kohonen, "The Self-Organizing Map", *Proceedings IEEE*, Vol. 78, No. 9, Sept 1990, pp 1464-1479.

[2] T. Kohonen, *Self-Organizing Maps*, Berlin: Springer-Verlag, 1995.

[3] G. Ferrate et al. (eds.), *Syntactic and Structural Pattern Recognition*, NATO ASI Series, Vol. F45, Berlin: Springer-Verlag, 1988.

[3] V. Neagoe, "A Circular Kohonen Network for Image Vector Quantization", *Parallel Computing: State-of - the Art and Perspectives* (E. H. D'Hollander, G. R Joubert, F. J. Peters and D. Trystram eds.), Vol. 11, Amsterdam-New York: Elsevier, 1996, pp. 677-680.

[5] V. Neagoe, O. Stanasila, *Recunoasterea formelor si retele neurale - algoritmi fundamentali* (Pattern Recognition and Neural Networks -Fundamental Algorithms), Bucharest: Ed. Matrix Rom, 1999.

[6] V. Neagoe, O. Stanasila, *Teoria recunoasterii formelor* (Pattern Recognition Theory), Bucharest: Ed. Academiei Romane, 1992.

[7] V. Neagoe, O. Cula, "A Fuzzy Connectionist Approach to Vowel Recognition", in *Real World Applications of Intelligent Technologies, part II*, (B. Reusch and D. Dascalu eds.), Bucharest: printed by National Institute for Research and Developmnt in Microtechnologies, 1998, pp. 48-52.

[8] V.Neagoe:"A Neural Error Correcting Approach Based on Levensthein Metrics for Automatic Romanian-English Translation." In: *Proceedings of the World Multiconference on Systemics, Cybernetics and Informatics*, July 22-25, 2001, Orlando, Florida, USA. Vol. XIII. ISBN: 980-07-7553-6. (2001) pp.14-17.

[9] V. Neagoe, " Metoda si sistem pentru traducerea automata utilizand o retea neuronala cu auto-organizare ", brevet de inventie nr. 116684/2001, OSIM, Romania.