Annales UMCS Informatica AI X, 2 (2010) 13-19
DOI: 10.2478/v10065-010-0048-2

Annales UMCS
Informatica
Lublin-Polonia
Sectio AI
http://www.annales.umcs.lublin.pl/

Kohonen networks application in speech analysis algorithms

Ireneusz Codello*, Wiesława Kuniszyk-Jóźkowiak, Adam Kobus

Institute of Computer Science, Maria Curie-Skłodowska University
pl. M.Curie-Skłodowskiej 1, 20-031 Lublin, Poland.

Abstract – This article presents the Kohonen network application in the speech analysis. The Authors have modified the traditional Kohonen network learning process like weights initialization, neurons reduction and neurons sorting. The results will be presented using authors' program – "WaveBlaster".

1 Introduction

The Kohonen network (or "self-organizing map", or SOM, for short) has been developed by Teuvo Kohonen. The basic idea behind the Kohonen network is to setup a structure of interconnected processing units ("neurons") which compete for the signal. While the structure of the map may be quite arbitrary, we are using rectangular maps.

On Fig. 1 we have:

- $SizeX * SizeY = K$ – number of Kohonen network's neurons,
- n – dimension of input vectors,
- each input vector $\overline{X}$ has n elements $X = \{x_1, x_2, \ldots, x_n\}$,
- each neuron y_i has exactly n connections, each one connected to consecutive elements x_i of $\overline{X}$,
- each element x_i is connected to all K neurons, so we have Kxn connections. Every connection is represented by it's weight $w_{ij}, i = 1 \ldots n, j = 1 \ldots K$.

As we can see, each node (connection) of the map is defined by a vector $\overline{W}_j = w_{ij}$ whose elements are adjusted during the training.

The basic training algorithm is quite simple:

*irek.codello@gmail.com

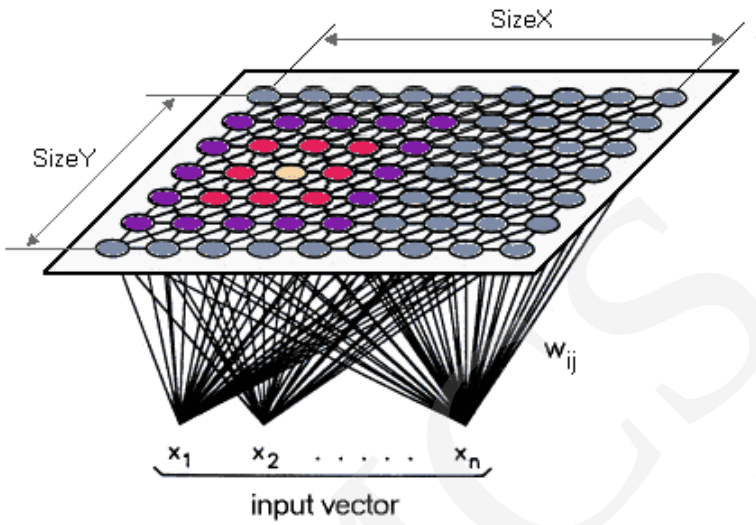


Fig. 1. Exemplary Kohonen network.

(1) Select the input vector from the training set – we could take consecutive vectors, or take them randomly.
(2) Find the neuron which is closest to the given input vector (i.e. the distance between $\overline{W}_j$ and $\overline{X}$ is a minimum). The metric can be arbitrary, usually Euklides', where the distance between the input vector and i-th neuron is defined:

$$d_j = \sqrt{\sum_{j=1}^{n}(x_j - w_{ij})^2}. \tag{1}$$

(3) Adjust the weight vectors of the closest node and the nodes around it in a way that the $\overline{W}_j$ move towards the training data:

$$\overline{W}_j = \overline{W}_j + \alpha(\overline{X} - \overline{W}_j) \tag{2}$$

(4) Repeat from step 1 for a fixed number of repetitions.

As a result of such learning, we can say that, in a Kohonen map, neurons located physically next to each other will correspond to classes of input vectors that are likewise next to each other (Fig. 2). Therefore such regions are called maps.

2 Speech analysis

The most important for our research is influences recognition. This process can divided into several steps.

First we need to change input speech signals into some set of parameters – for that purpose we usually use fallowing algorithms: Fourier Transform, FFT with octave filters, Linear

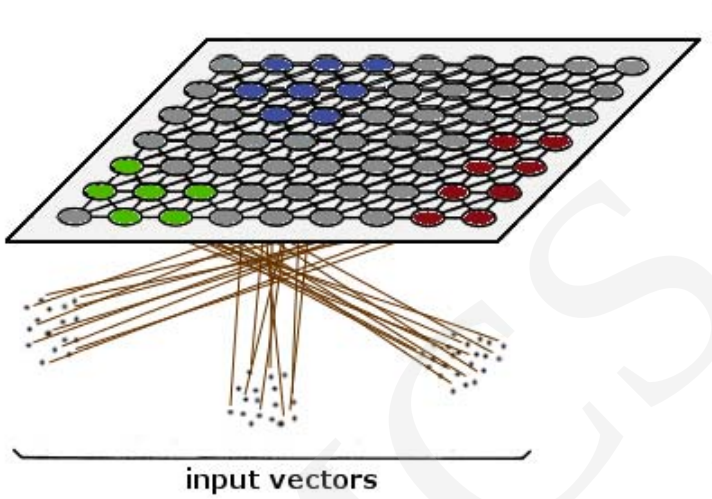
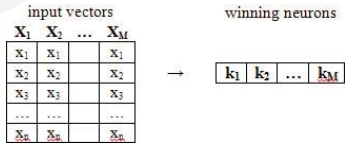


Fig. 2. Input vectors with corresponding maps on Kohonen network. Similar vectors are assigned to different neurons within one map.

Prediction, Continuous Wavelet Transform. Most of those transforms produce 2D results which are usually not suitable for us. We often use Neural Networks, like perceptron, to which we pass results of those transforms, therefore we need 1D data. And that is where we use Kohonen network. We pass the 2D transform's result into Kohonen network and we take only the winning neuron for each input vector. For input data we obtain only 1D winning neuron contour.



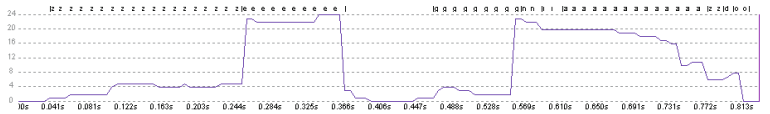


Fig. 3. Winning neuron contour of the utterance "że gniazdo". Kohonen network size: 5x5.

Then we cut such a contour on frames, for instance 800ms, and pass to other algorithms (like perceptron).

3 Kohonen learning process – Our approach

We added the fallowing modifications to Kohonen network learning process:

(1) Initialization.
After initializing the neurons using standard way: random values for the range <0,1> or <-1,1> with normal or Gauss distribution we sort them diagonally by theirs energy:

$$E_j = \sqrt{\sum -j = 1^n w_{ij}^2}. \tag{3}$$

Neurons which weights have less energy goes to top-left corner, and the ones with more energy to bottom-right corner.

1	2	4
3	5	7
6	8	9

It often happens that winning neurons of similar input vectors are spread all over the network, while we would like to have one, consistence map. We have observed during our research that this initialization considerably reduces this unwilling effect.

(2) Neurons reduction (post-learning).
This step is applied after the networked is learned using the standard algorithm described in Introduction. The purpose is to reduce each map (which contains similar neurons) to only one neuron within a map. We do the following:

- Find two closest neurons k_A, k_B (the distance between neurons weights is measured using Euklides' metric – equation 1).
- If the distance is less then some threshold (algorithm's parameter), fill weights of one of the neurons with zeros.
- Repeat steps 1. and 2. until exists pair of neurons closer then the threshold.

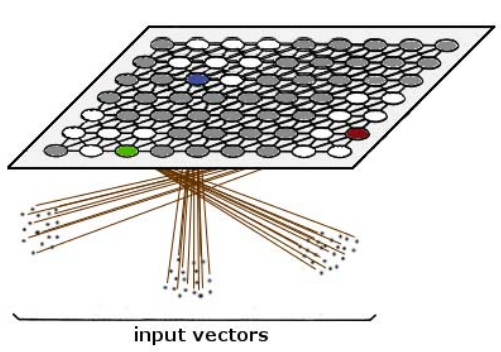


Fig. 4. Input vectors with corresponding maps on Kohonen network after "neurons reducing". Similar vectors are assigned to only one neuron. Other neurons (white ones) within maps are cleared.

The result of the reducing procedure is presented on Fig. 4., Fig. 7 and Fig. 8. As we can see, such a result is much clearer and therefore more useful than raw, unmodified result (see Fig. 6).

(3) Neurons sorting (post-learning).
Sometimes (it depends on what we want to recognize), at the end of the whole process, we sort the network's neurons. It sometimes increases the recognition ratio. The sorting procedure can be the same as in step 1. 'Initialization' but we also use another method. Instead of sorting neurons by energy, we sort them by 'similiarity'.
- Into top-left corner we place a neuron (k_1) with the least energy.
- Then we find the closest neuron (k_2) to k_1 (accordingly to equation 1.)
- Then we find the closest neuron (k_3) to k_2 and so on.

The exemplary results are presented on Fig. 5 – 8.

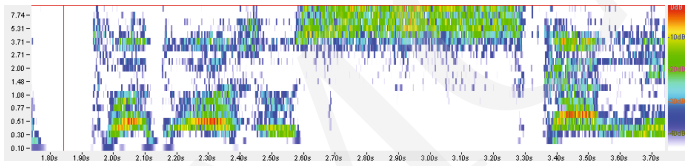


Fig. 5. Continuous Wavelet transform of the utterance with prolongation "sss". Screenshot from program "WaveBlaster".

Fig. 6. Winning neuron contour of the utterance from Fig. 5. Kohonen network size: 5x5. Screenshot from program "WaveBlaster"

Fig. 7. Winning neuron contour from Fig. 6 after "neurons reduction". Screenshot from program "WaveBlaster".

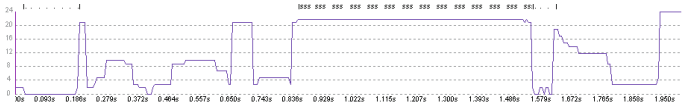


Fig. 8. Winning neuron contour from Fig. 7 after "neurons sorting". Screenshot from program "WaveBlaster".

4 WaveBlaster

All our research and all conclusions presented in this article were possible to make thanks to Our program – 'WaveBlaster' and it's quite developed set o parameters (Fig. 9). The presented way of post-learning is quite unique, therefore we had to create our own tool for this purpose.

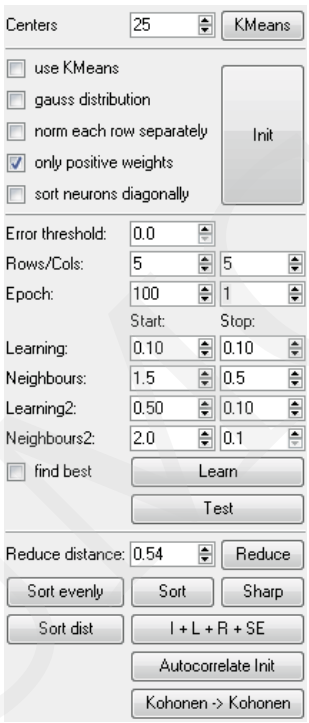


Fig. 9. Kohonen network option panel – screenshot from program WaveBlaster.

5 Summary

Kohonen network is a very powerful tool which can be applied to various problems, also to speech analysis. We find it very useful in algorithms for disorders recognition in speech, especially in reducing dimension of data passed into perceptrons (from 2D to 1D). We additionally improved our results by applying changes described in this article to Kohonen network learning procedure Thou, the described modifications of learning process were made in terms of increasing influence recognition ratio, we think that their nature is so general that they can be also applied to different problems.

References

[1] Garfield S., Elshaw M., Wermter S., Self-orgazizing networks for classification learning from normal and aphasic speech, The 23rd Conference of the Cognitive Science Society (Edinburgh, 2001): 187–196.

[2] Kohonen T., Self-organizing maps 34 (2001): 2173–2179.

[3] Szczurowska I., Kuniszyk-Jóźkowiak W., Smołka E., The application of Kohonen and multilayer perceptron network in the speech nonfluency analysis, Archives of Acoustics 206(31)(4 Supplement): 205–210.

[4] Szczurowska I., Kuniszyk-Jóźkowiak W., Smołka E., Application of artificial neural networks in speech nonfluency recognition, Polish Jurnal of Environmental Studies 16(4A) (2007): 335–338.