BIOMEDICAL IMAGE ANALYSIS USING SELF-ORGANIZING MAPS

L. Grajciarová, J. Mareš, P. Dvořák, A. Procházka

Department of Computing and Control Engineering,
Institute of Chemical Technology, Technická 5, 166 28 Prague

Abstract

The implementation of self organizing map (SOM) and its application to the analysis of biomedical images are presented. The SOM algorithm was implemented in MATLAB program suite with various optional parameters enabling the adjustment of the model according to the user's requirements. For easier application of SOM the graphical user interface has been developed. The segmentation and edge detection procedures are critical steps in the analysis of biomedical images, enabling for instance the detection of the abnormal structure or the recognition of different types of tissue. The self-organizing map provides a quick and easy approach for these tasks with satisfying quality of outputs, which has been verified using the high-resolution CT images capturing the expressions of the Granulomatosis with polyangiitis (GPA) disease.

1 Introduction

The self-organizing map (SOM) [7, 12] is widely applied approach for clustering and pattern recognition that can be used in many stages of the image processing, e. g. in color image segmentation [10], generation of a global ordering of spectral vectors [14], image compression [13], binarisation document [4] or texture edge detection [9] etc.

In the contribution, the implementation of SOM in MATLAB is presented and the verificaion of the abilities of the implemented SOM for analysis of biomedical images is showed.

2 Software Description

The SOM algorithm was implemented in MATLAB program suite [2] with various optional parameters enabling the adjustment of the model according to the user's requirements. For easier application of SOM the graphical user interface (GUI) was developed, see Figure 1. [1]

2.1 Basic Settings

A SOM has two layers of neurons, see Figure 2. The input layer (size Nx1) represents input data *x1*, *x2*, , *xM* (M inputs, each input is N dimensional). The output layer (size KxL), that may have a linear or 2D arrangement, represents clusters in which the input data will be grouped. Each neuron of the input layer is connected with all neurons of the output layer through the weights W (size of the weight matrix is KxLxN)

Using GUI the user can set the structure of network, i. e. determine the location of the file with input data and define the size of the output layer of SOM, see Figure 1 part 1.

2.2 Learning

A SOM is neural network with unsupervised type of learning, i. e. no cluster values denoting an a priori grouping of the data instances are provided.

The learning process is divided in epochs, during which the entire batch of input vectors is processed. The epoch involves the following steps:

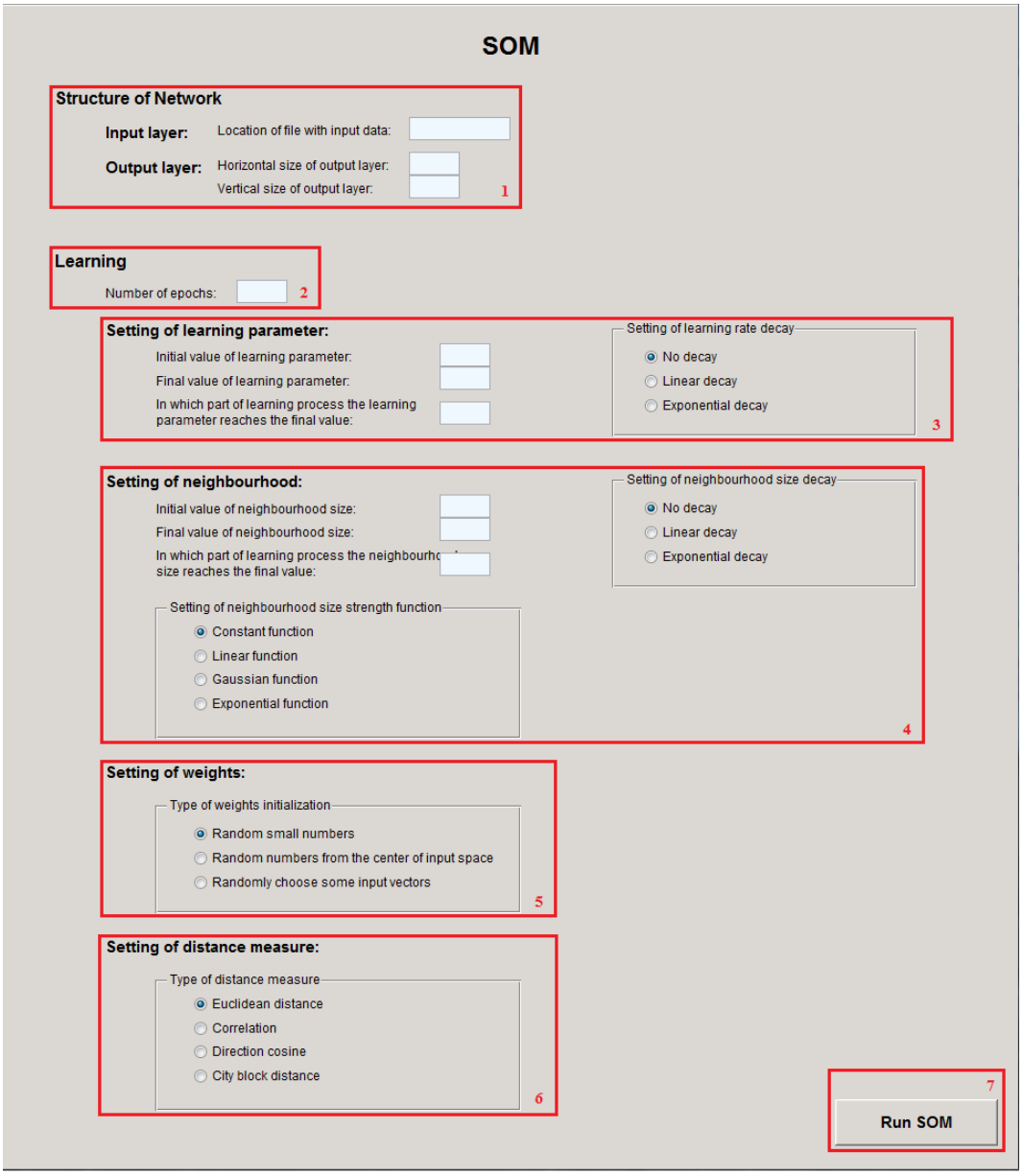


Figure 1: The graphical user interface of the implemented SOM: **1** The basic settings of SOM, **2** The selection of the number of epochs, **3** The settings of learning parameter, **4** The settings of neighbourhood, **5** The Setting of weights, **6** The Setting of distance measure, **7** The 'Run SOM' button

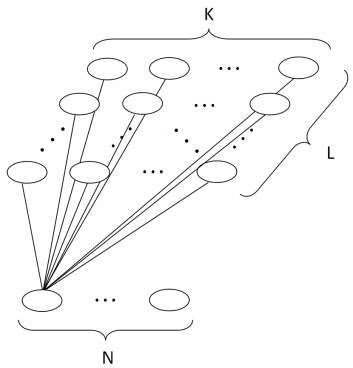


Figure 2: Structure of SOM

1. Consecutive submission of an input data vector to the network.
2. Calculation of a distances between the input vector and the weight vectors of the neurons of the output layer.
3. Selection of the nearest (the most similar) neuron of the output layer to the presented input data vector.
4. An adjustment of the weights.

The weights adapt during the learning process based on a competition, i. e. the nearest (the most similar) neuron of the output layer to the submitted input vector becomes a winner and its weight vector and the weight vectors of its neighbouring neurons are adjusted according to

$$\mathbf{W} = \mathbf{W} + \lambda \, \phi_s \, (\mathbf{x_i} - \mathbf{W}), \tag{1}$$

where $\mathbf{W}$ is the weight matrix, $\mathbf{x_i}$ the submitted input vector, λ the learning parameter determining the strength of the learning and ϕ_s the neighbourhood strength parameter determining how the weight adjustment decays with distance from the winner neuron (it depends on s, the value of the neighbourhood size parameter).

Using GUI the user can set the parameters of the learning process including the number of epochs, see Figure 1 part 2.

2.2.1 Learning Parameter

The learning parameter, corresponding to the strength of the learning, is usually reduced during the learning process. It decays from the initial value to the final value, which can be reached already during the learning process, not only at the end of the learning. There are several common forms of the decay function see Figure 3:

1. No decay
$$\lambda_t = \lambda_0, \tag{2}$$
2. Linear decay
$$\lambda_t = \lambda_0 \left(1 - \frac{t}{\tau}\right), \tag{3}$$
3. Gaussian decay
$$\lambda_t = \lambda_0 \mathrm{e}^{-\frac{t^2}{2\tau^2}}, \tag{4}$$
4. Exponential decay
$$\lambda_t = \lambda_0 \mathrm{e}^{-\frac{t}{\tau}}, \tag{5}$$

where T is total number of iterations, λ_0 and λ_t are the initial learning rate and that at iteration t, respectively. The learning parameter should be in the interval $< 0.01, 1 >$.

Figure 1 part 3 shows options of the GUI regarding the learning parameter.The initial and final values of learning parameter have to be set. The initial value should be close to 1, the final value should be small, but not smaller than 0.1. Simultaneously, a point in the learning process in which the learning parameter reaches the final value has to be determined. It is represented as a number between 0 and 1. The learning rate decay has to be set as well.

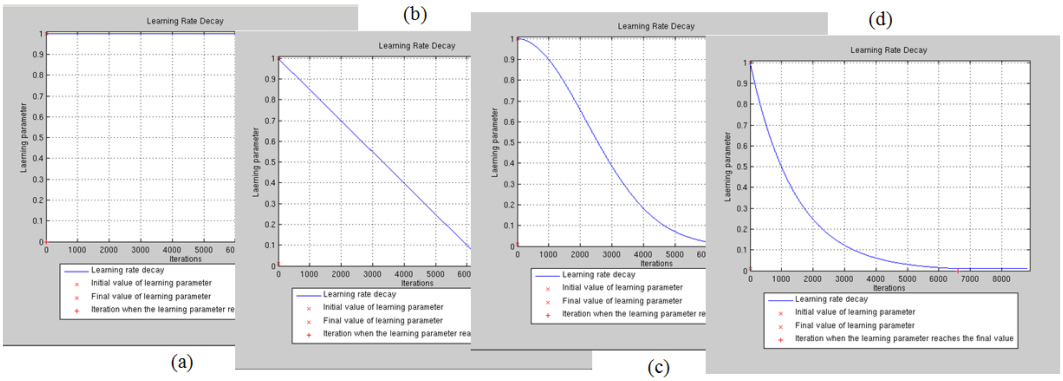


Figure 3: Learning rate decay function (dependence of the learning parameter on the number of iterations): a) No decay, b) Linear decay, c) Gaussian decay, d) Exponential decay

2.2.2 Neighbourhood

In SOM learning not only the winner but also the neighbouring neurons adjust their weights. It produces topology preservation. There several ways to define a neighbourhood (see Figure 4). All neighbour weight vectors are shifted towards the presented input vector, however, the winning neuron update is the most pronounced and the farther away the neighbouring neuron is, the less its weight is updated. The neighbourhood strength function determines how the weight adjustment decays with distance from the winner. The neighbourhood size function determines how the size of neighbourhood decays with increasing number of iterations.

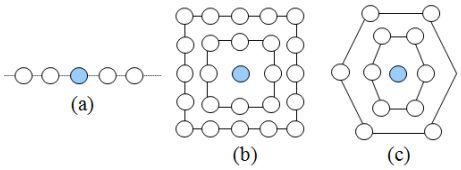


Figure 4: Types of neighbourhood: a) Linear arrangements, b) Square arrangements, c) Hexagonal arrangements

Figure 1 part 4 shows options of the GUI regarding the neighbourhood. The initial and final values of the neighbourhood size have to be set. The initial value can be up to the size of the output layer, the final must not be less than 1. That point in the learning process has to be determined, in which the neighbourhood size reaches the final value, i. e. number between 0 and 1. The neighbourhood size decay has to be set as well.

2.2.3 Weights

The implemented SOM is trained in recursive mode, i. e. the weights of the winning neurons are updated after each insertion of an input vector. The user has to choose the type of the weights initialization, see Figure 1 part 5 and Figure 5.

2.2.4 Distance Measures

The criterion for victory in the competition of the neurons of the output layer, i. e. the measure of the distance between the presented input vector and its weight vectors, may have many forms. The most commonly used are:

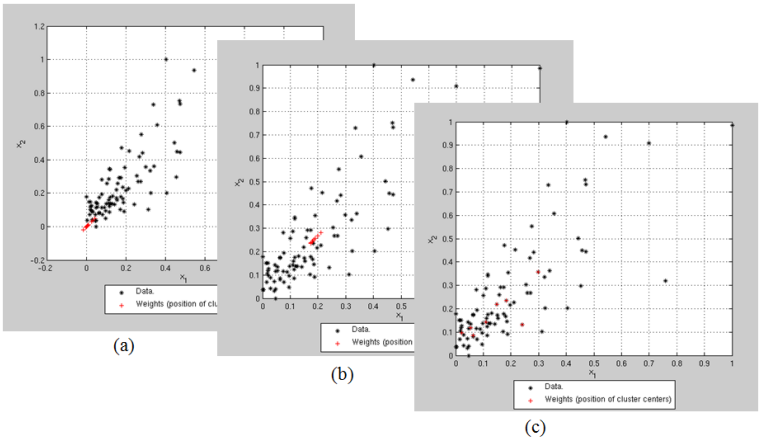


Figure 5: Weight vectors initialization: a) Random small numbers, b) Vectors near the center of gravity of inputs, c) Randomly chosen some input vectors as initial weight vectors

1. Euclidean distance

$$d_j = \sqrt{\sum_{i=1}^{N} (x_i - w_{ji})^2}, \tag{6}$$

2. Correlation

$$d_j = \sum_{i=1}^{N} \frac{(x_i - \overline{x})(w_{ji} - \overline{w_j})}{\sigma_x \sigma_{w_j}}, \tag{7}$$

3. Direction cosine

$$d_j = \frac{\sum_{i=1}^{N} x_i w_{ji}}{\|x_i\| \|w_{ji}\|}, \tag{8}$$

4. Block distance

$$d_j = \sum_{i=1}^{N} |x_i - w_{ji}|, \tag{9}$$

where x_i is i-th component of the input vector, w_{ji} i-th component of the j-th weight vector, N dimension of the input and weight vectors, $\overline{x}$ mean value of the input vector x, $\overline{w_j}$ mean value of the weight vector w_j, σ_x standard deviation of the input vector x, σ_{w_j} standard deviation of the weight vector w_j, $\|x_i\|$ length of the input vector x and $\|w_{ji}\|$ length of the weight vector w_j.

The implemented SOM offers several measures of the closeness of a weight vector to an input vector, which the user can select, see Figure 1 part 6).

2.2.5 Results of SOM

The *learning progress criterion*, minimized over the learning process, is the sum of distances between all input vectors and their respective winning neuron weights, calculated after the end of each epoch, according to

$$D = \sum_{i=1}^{k} \sum_{n \in c_i} (\mathbf{x_n} - \mathbf{w_i})^2, \tag{10}$$

where $\mathbf{x_n}$ is the n-th input vector belonging to cluster c_i whose center is represented by $\mathbf{w_i}$ (e. i. the weight vector of the winning neuron representing cluster c_i).

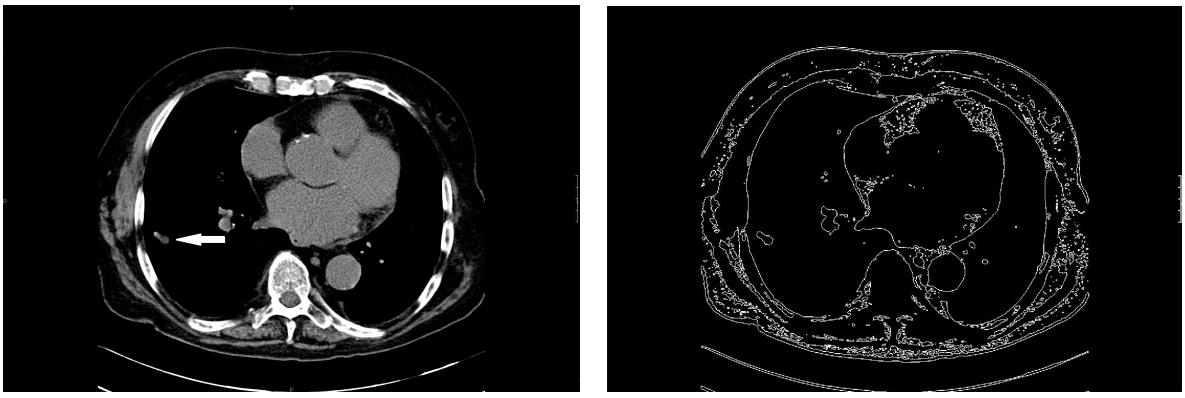
The weight adjustment corresponding to the smallest learning progress criterion is the result of the SOM learning process. These weights represent the cluster centers.

For the best result, the SOM should be run several times with various settings of SOM parameters to avoid detection of local minima and to find the global optimum on the error surface plot.

Pressing the button 'Run SOM' in the GUI, the SOM starts to run, see Figure 1 part 7.

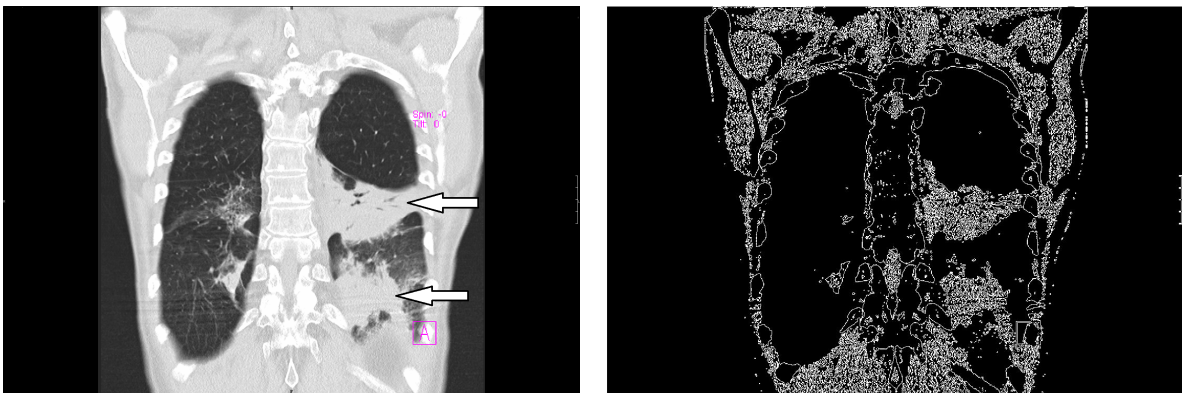
3 Software Verification

The verification of the software was based on detection of all three expression forms of the GPA disease [6, 8, 3, 11] in high-resolution CT images (provided by Department of Nephrology, First Faculty of Medicine and General Faculty Hospital, Prague, Czech Republic), see Figure 6, 7, 8. The obtained results were discussed with an expert that confirmed the suitability of the software for analysis of images with GPA expressions. For more information about the study see [5].



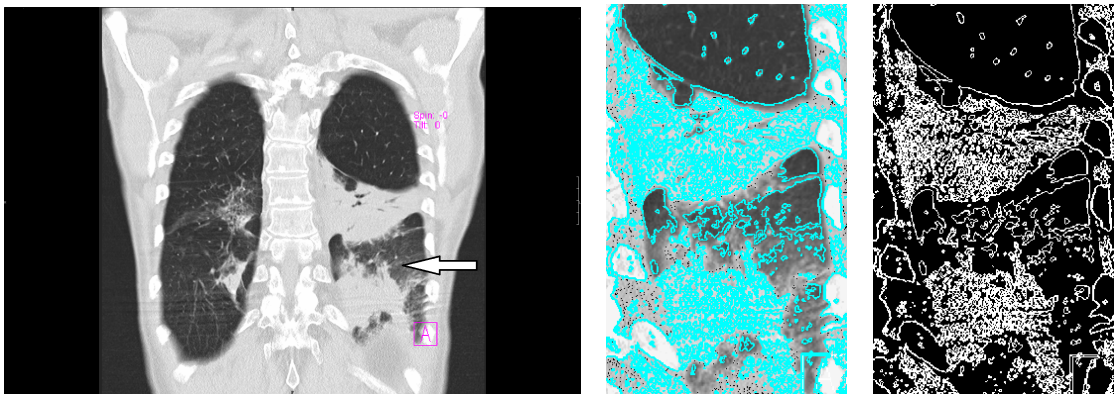
(a) (b)

Figure 6: Detection of granuloma. (a) Transverse high-resolution CT image of both lungs with active granulomatosis (white arrow). b) The edge detection result obtained by the SOM. The granulomatosin is detected with sufficient accuracy



(a) (b)

Figure 7: Detection of masses. (a) Coronal high-resolution CT image of both lungs with masses (white arrows). Possibility of the detection of the masses is aggravated by the'ground-glass surrounding the lower part of the first mass and the upper part of the second mass. The artifacts originated by coughing movements of the patient makes the detection process difficult as well. (b) The edge detection result obtained by the SOM. The'masses are detected and distinguished from the ground-glass with sufficient accuracy



(a) (b) (c)

Figure 8: Detection of ground-glass. (a) Coronal high-resolution CT image of both lungs with ground-glasses (white arrows). Possibility of the detection of the ground-glasses is complicated by the masses in close proximity to the ground-glasses. The artifacts originated by coughing movements of the patient makes the detection process hard as well. (b) The edge detection result obtained by the SOM. The ground-glasses are detected and distinguished from the masses. (c) The overlap of the original CT image and the edge detection result (cyan color)

4 Acknowledgements

The work was supported by the specific university research MSMT No. 21/2012 and the research grant PRVOUKP25/LF1/2.

References

[1] Biomedical Data Analysis Using Self-Organizing Maps [cited 2012 Oct 23]. Available from: http://uprt.vscht.cz/vav/SoftwareRealizace2011.htm.

[2] *MATLAB, version 7.11.0 (R2010b).* The MathWorks Inc., Natick, Massachusetts, 2010.

[3] L. Annanthakrishan, N. Sharma, and J. P. Kanne. Wegener's granulomatosis in the chest: High-resolution ct findings. *AJR Am J Roentgenol*, 192:676–82, 2009.

[4] E. Badekas and N. Papamarkos. Document binarisation using kohonen som. *IET Image Processing*, 1:67–84, 2007.

[5] L. Gráfová, J. Mareš, A. Procházka, and P. Konopásek. *Artificial Neural Networks: Edge Detection in Biomedical Images Using Self-Organizing Maps.* InTech, 2012.

[6] J. C. Jennette. Nomenclature and classification of vasculitis: lessons learned from granulomatosis with polyangiitis (wegener's granulomatosis). *Clin Exp Immunol.*, 164 Suppl 1:7–10, 2011.

[7] T. Kohonen. *Self-Organization and Associative Memory.* Springer-Verlag, 1989.

[8] S. E. Lane, R. Watts, and D. G. I. Scott. Epidemiology of systemic vasculitis. *Curr Rheumatol Rep*, 7:270–275, 2005.

[9] Jyh-Charn Liu and Gouchol Pok. Texture edge detection by feature encoding and predictive model. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 2, page 1105 1108, 1999.

[10] J. Moreira and L. Da Fontuora. Neural-based color image segmentation and classification. *Anais do*, IX SIBGRAPI:47–54, 1996.

[11] Y. Renaudineau and Y. Le Meur. Renal involvement in wegener's granulomatosis. *Clinic Rev Allerg Immunol*, 35:22–29, 2008.

[12] S. Samarasinghe. *Neural Networks for Applied Sciences and Engineering: From Fundamentals to Complex Pattern Recognition.* Auerbach Publications, 2006.

[13] D. K. Sharma, L. Gaur, and D. Okunbor. Image compression and feature extraction using kohonen's self-organizing map neural network. *Journal of Strategic E-Commerce*, 5:25–38, 2007.

[14] P. J. Toivanen, J. Ansamki, J. P. S. Parkkinen, and J. Mielikinen. Edge detection in multispectral images using the self-organizing map. *Pattern Recognition Letters*, 24:2987–2994, 2003.

Lucie Grajciarová
grafoval@vscht.cz

Jan Mareš
maresj@vscht.cz

Petr Dvořák
dvorake@vscht.cz

Aleš Procházka
prochaz@vscht.cz