Self-Organising Maps for Classification with Metropolis-Hastings Algorithm for Supervision.

Piotr Płoński[1] and Krzysztof Zaremba[1]

[1]Institute of Radioelectronics, Warsaw University of Technology,
Nowowiejska 15/19,00-665 Warsaw, Poland,
{pplonski,zaremba}@ire.pw.edu.pl

Abstract. Self-Organising Maps (SOM) provide a method of feature mapping from multi-dimensional space to a usually two-dimensional grid of neurons in an unsupervised way. This way of data analysis has been proved as an efficient tool in many applications. SOM presented by T.Kohonen originally were unsupervised learning algorithm, however it is often used in classification problems. This paper introduces novel method for supervised learning of the SOM. It is based on neuron's class membership and Metropolis-Hastings algorithm, which control network's learning process. This approach is illustrated by performing recognition tasks on nine real data sets, such as: faces, written digits or spoken letters. Experimental results show improvements over the state-of-art methods for using SOM as classifier.

Keywords: Self-Organising Maps, Classification, Supervised learning, Metropolis-Hastings algorithm

1 Introduction

Self-Organising Map (SOM) is a neural network presented in 1982 by T.Kononen [11]. Due to human readability of the model, easy implementation and fast learning, SOM gained great popularity in many data analysis problems [13]. SOM was originally presented as an unsupervised algorithm, however there are extensions that enable to use SOM as a classifier. They can be generally divided into three groups.

The first group of methods is based on class membership of each neuron. In this approach, SOM is first learned in unsupervised manner. After training, class membership is found for each neuron, based on sample's class label. Class membership can be crisp or fuzzy [18], [20], [9], [8]. In the testing phase, the simplest approach predicts the class based on the class of winning neuron - so-called 'winner-takes-all method' (WTA). There are also more sophisticated methods based for example on k-Nearest Neighbour rule or interpretation of weights[21].

The second approach combines class vector in binary coded manner with attribute vector during learning process. In the testing phase, only the attribute vector is presented to the SOM. Sample's class is denoted, based on neurons

weights corresponding to the class. There are several approaches for doing this [16], [2], [22], [9], [19]. However, the [16] algorithm seems to present the most generalized approach from those.

The third technique for using SOM as a classifier is to use as many SOM networks as number of classes. This approach is well known from Learning Vector Quantization (LVQ) algorithm [12]. In the simplest way, each network is trained on samples from corresponding class [7], [4]. In more complex approach [12], the network is trained on samples from both corresponding and other classes.

Method presented in this paper combines first and third approach.It uses Metropolis-Hastings (MH) algorithm [15], [6] and class membership of neurons to control neurons participation in the training process. The MH is well know from Simulated Annealing (SA) [10] algorithm. There were several attempts to use MH [17] or SA [5], [3] in SOM. However, they were focused on weights optimization rather than boosting SOM's classification performance.

2 Methods

Let's denote data set as $D = \{(\boldsymbol{x_i}, c_i)\}$, where $\boldsymbol{x_i}$ is an attribute vector, $\boldsymbol{x} \in \mathcal{R}^d$ and c_i is a discrete class number of i-th sample, $i = [1, 2, ..., N]$ and $c = [1, 2, ..., C]$. Sometimes the class number will be encoded as a binary vector and denoted as $\boldsymbol{y_i}$, where $\boldsymbol{y_{ij}} = 1$ for $j = c_i$ and $\boldsymbol{y_{ij}} = 0$ otherwise.

2.1 Unsupervised Learning SOM Algorithm

Herein, we used SOM as a two-dimensional grid of neurons. Each neuron is represented by a weight vector W_{pq}, where (p, q) are indexes of the neuron in the grid. In the learning phase all samples are shown to the network in one epoch. For each sample we search for a neuron which is closest to the i-th sample. The distance is computed by:

$$Dist_{train}(D_i, W_{pq}) = (\boldsymbol{x_i} - W_{pq})^T(\boldsymbol{x_i} - W_{pq}). \tag{1}$$

The neuron (p, q) with the smallest distance to i-th sample is called the Best Matching Unit (BMU), and we note its indexes as (r, v). Once the BMU is found, the weight update step is executed. The weights of each neuron are updated with the following formula:

$$W_{pq}(t+1) = W_{pq}(t) + \eta(W_{pq}(t) - \boldsymbol{x_i}), \tag{2}$$

where t is an iteration number and η is a learning coefficient. It can be written as $\eta = \mu\tau$, where μ is the size of the learning step and τ is the neighbourhood function. Learning step size is decreased between consecutive epochs, so that network's ability to remember patterns is improved. It is described by $\mu = \mu_0 exp(-e\lambda_\mu)$, where μ_0 is the initial step size, e is the current epoch number and λ_μ is responsible for regulating the speed of the decrease. Neighbourhood function controls changing of the weights with respect to the distance to the

BMU. It is noted as $\tau(r, v, p, q) = exp(-\alpha((r-p)^2+(v-q)^2))$, where α describes the neighbourhood function width. This parameter is increasing during learning $\alpha = \alpha_0 exp(-(e_{stop} - e)\lambda_\alpha)$ - it assures that neighbourhood becomes narrower during training. Network is trained till chosen number of learning procedure epochs e_{stop} is exceeded.

2.2 SOM-WTA

From the first group of methods we will use SOM in WTA configuration (SOM-WTA). After unsupervised training process, where BMU for each sample is found, the class membership for each neuron is computed. BMU contains class number of the matching samples. After presentation of all the samples, each neuron's class membership is decided based on major class number. In the testing phase, the class of an input sample is assigned based on the class of the computed BMU. The main disadvantage of this method are so-called 'empty neurons', when neuron has never been selected as BMU during training but is selected in the testing[21].

2.3 SOM-LASSO

The second approach used in this paper is the so-called 'Learning Associations by Self-Organisation' (SOM-LASSO), first described in [16]. During the learning phase, additionally to attributes it takes into consideration the class vector $\boldsymbol{y_i}$. Each neuron contains part of weights corresponding to the attributes W^x_{pq} and a class vector W^y_{pq}, so $W_{pq} = [W^x_{pq}; W^y_{pq}]$. The measure of the distance used during training is computed by:

$$Dist_{train}(D_i, W_{pq}) = (\boldsymbol{x_i} - W^x_{pq})^T(\boldsymbol{x_i} - W^x_{pq}) + (\boldsymbol{y_i} - W^y_{pq})^T(\boldsymbol{y_i} - W^y_{pq}). \quad (3)$$

The rest of the training process is the same as in original SOM. In testing, the exploitation phase is performed, where only the part with attributes is presented to the network. The BMU is found by computing a distance between an attribute input vector and an attribute part of the weights, using the following formula:

$$Dist_{test}(D_i, W_{pq}) = (\boldsymbol{x_i} - W^x_{pq})^T(\boldsymbol{x_i} - W^x_{pq}). \quad (4)$$

For the tested sample, the designated class corresponds to position of maximum value in the part which codes class information W^y_{pq} in BMU weights.

2.4 SOM-SNEC

The third method uses separate network for each class, we called it SOM-SNEC. Each network is learned only with samples from the corresponding class in an unsupervised manner. In the testing process, the BMU is computed in each network. Sample's class is designated from the network with the closest BMU. The main disadvantage of this method is that it losts possibility to visualize all samples on a single map.

2.5 Proposed Method (SOM-MH)

In this method, neuron's class membership is described by probability. We note $P_{pq}(h)$ as probability of neuron's membership in class number h, where (p, q) are neuron's indexes. For each training iteration[1] only selected group of neurons will take part in the training. Selection is described by a matrix T, where $T^i_{pq} = 1$ means that neuron (p, q) will participate in learning using i-th sample, $T^i_{pq} = 0$ otherwise. Neurons are selected in two steps. First choose neurons having maximum probability for the class matching the class c_i of the input sample:

$$T^{i(1)}_{pq} = \begin{cases} 1 & \text{if } \arg\max_h(P_{pq}(h)) = c_i; \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

In the second step, remaining neurons are considered, with $T^{i(1)}_{pq} = 0$. The decision on joining the training with i-th sample is taken upon MH algorithm. The probability of joining is computed using following equation:

$$J^i_{pq} = 1 - exp(-\rho P_{pq}(c_i) e_{stop}/e), \tag{6}$$

where ρ is the parameter that controls the number of neurons selected additionally to learning in the MH step, $\rho \in [0, 1]$. The fact that number of epochs e is presented in eq.(6) ensures that neurons added during MH step will be selected less frequently at the end of learning process than at its beginning. This can be interpreted as a hesitation of the neuron, which decreases during the training. Whether the MH decision will be positive, we draw random number a from an uniform distribution, $a \in [0, 1]$. The neuron will be added to the training group if a is smaller than J^i_{pq}:

$$T^{i(2)}_{pq} = \begin{cases} 1 & \text{if } a < J^i_{pq}; \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

This procedure is repeated for each sample. The final decision on neuron selection is a logical 'or' of the decisions $T^i_{pq} = T^{i(1)}_{pq} \vee T^{i(2)}_{pq}$.

After each epoch new probabilities are updated. During training for each i-th sample the neighbourhood value τ_i is added to the neuron's probability of membership in a given class:

$$P'_{pq}(h) = \sum_i^N T^i_{pq} \tau_i, \text{ for } h = c_i. \tag{8}$$

The neighbourhood value τ_i represents the belonging of the neuron to the input sample's class. After all iterations in a given epoch, the probability are normalized and updated with formula:

$$P_{pq}(h) = \frac{P'_{pq}(h)}{\sum_{j=1}^C P'_{pq}(j)}. \tag{9}$$

[1] One iteration is a showing to the network one sample. One epoch is a showing to the network all samples.

3 Results

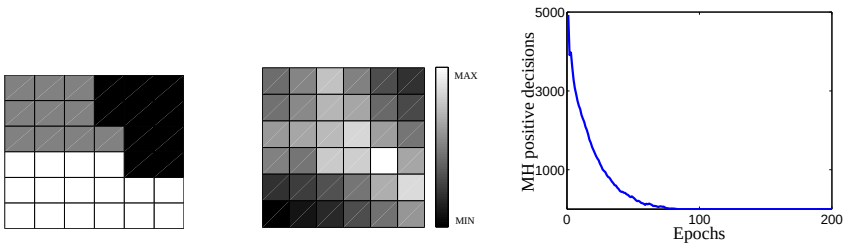
At the beginning we will present properties of proposed SOM-MH method, then we will compare it to the SOM-WTA, SOM-LASSO, SOM-SNEC and LVQ methods. The comparison is made on 9 real data sets. We used data sets 'Wine', 'Ionosphere', 'Iris', 'Isolet', 'Digits', 'Sonar', 'Spam', 'Pima' from the 'UCI Machine Learing Repository' [2] [1], and set 'Faces' are from the 'The ORL Database of Faces'[3]. In all experiments we used following parameters values: $e_{stop} = 200$, $\mu_0 = 0.1$, $\lambda = 0.0345$, $\alpha_0 = 0.1$, $\lambda_\alpha = 0.008$. All variants of SOM algorithms were implemented by authors in Matlab. The LVQ algorithm was used from Matlab Neural Networks Toolbox with default learning parameters and number of epochs e_{stop}.

	Train examples	Test examples	Attributes	Classes	Single net size	Multiple nets size	MH ρ
Faces	320	80	50*	40	15x16	2x3	0.005
Ionosphere	280	71	34	2	6x8	4x6	0.005
Iris	120	30	4	3	6x6	3x4	0.25
Isolet	6237	1560	100*	26	12x13	2x3	0.75
Digits	4496	1124	64	10	15x16	4x6	0.5
Wine	142	36	13	3	6x6	3x4	0.2
Pima	614	154	8	2	12x12	8x9	0.25
Sonar	166	42	60	2	8x9	6x6	0.1
Spam	3680	921	57	2	12x12	8x9	0.75

Table 1: Description of data sets used to test performance and parameters of networks. Single net size was used for methods SOM-WTA, SOM-LASSO and SOM-MH, multiple nets size is for SOM-SNEC and LVQ. (*) In 'Isolet' and 'Faces' data sets, the number of attributes was reduced with PCA.

To show SOM-MH algorithm properties, we learned 7x7 network with 'Iris' data set. Fig.3a presents network with neurons assigned to one of the three classes. Fig.3b presents cumulative number of positive MH decisions taken for each neuron during the whole training. We can observe that neurons which lay on the border between the different classes have higher number of positive MH decisions than neurons which have neighbour neuron from the same class. The highest number of positive MH decisions are for neuron which lay in the border of the three classes. Fig.3c presents number of positive MH decisions for network in each epoch for MH parameter $\rho = 0.5$. It can be observed that number of positive MH decisions are decreasing during learning, which can be interpreted as making the network more confident.

[2] http://archive.ics.uci.edu/ml/
[3] http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html



(a) (b) (c)

Fig. 1: Properties of SOM-MH network tested on 'Iris' data set, (a) network with neuron color presenting class membership, (b) network with neuron color presenting number of positive MH decisions taken during training, (c) number of positive MH decisions of all network taken in each training epoch.

Network sizes used for each data sets are presented in Table.1. For each method, the total number of used neurons are the same. For SOM-WTA, SOM-LASSO, SOM-MH for all the classes used a single network. For SOM-SNEC and LVQ, there are multiple networks, one for each class (and hence different column in Table 1). For SOM-MH the parameter ρ must be tuned. We checked several values of ρ, $\rho = \{1, 0.75, 0.5, 0.25, 0.2, 0.15, 0.1, 0.05, 0.005\}$ and for each data set an optimal value was selected by cross-validation. Selected ρ values are presented in Table 1. For each data sets we made 10 repetitions to avoid effect of local minima. As an accuracy measure, we take the percentage of incorrect classifications. The mean results for all the methods for training subsets are presented in Table 2.

	SOM-WTA	SOM-LASSO	SOM-SNEC	SOM-MH	LVQ
Faces	28.88±4.35	35.75±5.11	**3.75±2.04**	5.25±2.99	6.5±3.05
Ionosphere	14.23±4.06	13.66±4.2	10.85±3.45	**10.42±3.65**	13.1±3.93
Iris	6.67±4.97	6±3.06	3.33±2.22	**2±1.72**	6±4.66
Isolet	21.5±1.74	8.36±0.61	**5.96±0.45**	6.83±0.8	7.6±0.48
Digits	6.27±0.8	6.29±0.64	3.16±0.44	**3.02±0.44**	17.94±2.44
Wine	6.94±4.77	4.17±4.39	3.33±2.55	**2.74±2.27**	4.17±2.36
Pima	28.05±4.59	24.55±2.41	26.69±3.39	22.4±3.36	**21.56±3.77**
Sonar	36.19±6.99	24.76±5.96	24.05±4.69	**23.81±6.04**	26.67±6.53
Spam	16.35±1.02	12.74±1.25	12.42±1.17	**11.77±1.3**	37.74±1.34

Table 2: Percent of incorrect classification on testing subsets for the SOM-WTA, SOM-LASSO, SOM-SNEC, SOM-MH and LVQ methods. Results are mean and σ over 10 runs.

The poorest accuracy on almost all data sets was obtained by SOM-WTA method. This was expected, as this method does not use the information about sample's class during the tuning of the weights. However, this method was better than SOM-LASSO on the 'Faces' training set. Poor accuracy of SOM-LASSO on this set can be explained by comparable lengths of attribute and class vectors. The best accuracy on this set was obtained by SOM-SNEC method, which was also the best method on 'Isolet' data sets. On 'Pima' data set the LVQ method gives the best performance. On all other sets, the SOM-MH method gives the lowest incorrect classifications. If the comparison is made only for methods that use single SOM network, SOM-MH is significantly better than SOM-WTA and SOM-LASSO on all data sets. SOM-SNEC has similar results to SOM-MH. However, by using SOM-SNEC we lost important feature of SOM - the ability of data visualization on a single map.

4 Conclusions

A new method SOM-MH for using SOM as a classifier was presented. It uses neuron's class membership and Metropolis-Hastings algorithm to control neuron's learning process. This can be interpreted as simulating neuron's hesitation during the learning or as simulated annealing of class membership. The hesitation of neuron decrease during the learning. The proposed method was compared to other state-of-art methods for using SOM in classification tasks. Test results confirm that the proposed method improve accuracy of classification. The other supervised clustering algorithms can be improved with proposed method. Matlab implementation of the SOM-MH model is available at `http://home.elka.pw.edu.pl/~pplonski/som_mh`.

References

1. Asuncion, A., Newman, D.J.: UCI machine learning repository. University of California, Irvine, School of Information and Computer Sciences (2007)
2. Brereton, R.G.: Self organising maps for visualising and modelling. Chemistry Central Journal, vol.6 (2012)
3. Dozono, H., Tokushima, H., Hara, S., Noguchi, Y.: An Algorithm of SOM using Simulated Annealing in the Batch Update Phase for Sequence Analysis. In WSOM (2005) pp 171-178
4. Fessant, F., Aknin, P., Oukhellou, L., Midenet, S.: Comparison of Supervised Self-Organizing Maps Using Euclidian or Mahalanobis Distance in Classification Context. In IWANN(2001) pp 637-644
5. Fiannaca, A., Di Fatta, G., Gaglio, S., Rizzo, R., Urso, A.: Improved SOM Learning Using Simulated Annealing. In ICANN (2007) pp 279-288
6. Hastings, W.K.: Monte Carlo Sampling Methods Using Markov Chains and Their Applications. Biometrika vol.57, pp 97–109 (1970)
7. Hinton, G.E., Dayan, P., Revow, M.: Modeling the manifolds of images of handwritten digits, IEEE Transactions on Neural Networks vol.8, pp 65-74 (1997)

8. Hu, W., Xie, D., Tan, T., Maybank, S.: Learning Activity Patterns Using Fuzzy Self-Organizing Neural Network. IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics), vol.34, pp 1618-1626 (2004)
9. Kästner, M., Villmann, T.: Fuzzy Supervised Self-Organizing Map for Semi-supervised Vector Quantization. In ICAISC(2012) pp 256-265
10. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by Simulated Annealing. Science vol.220, pp 671–680 (1983)
11. Kohonen, T.: Self-organized formation of topologically correct feature maps. Biological Cybernetics, vol.43 pp 59-69 (1982)
12. Kohonen, T.: The Self-Organizing Map. Proceedings of the IEEE, vol.78, pp 1464-1480 (1990)
13. Kohonen, T., Oja, E., Simula, O., Visa, A., J. Kangas, Engineering applications of the self-organizing map. Proceedings of the IEEE(2002), vol. 84, no. 10, pp. 1358-1384.
14. Melssen, W., Wehrens, R., Buydens, L.: Supervised Kohonen networks for classification problems. Chemometrics and Intelligent Laboratory Systems, vol.83,pp 99–113 (2006)
15. Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E.: Equations of State Calculations by Fast Computing Machines. Journal of Chemical Physics vol.21, pp 1087–1092 (1953)
16. Midenet, S., Grumbach, A.: Learning Associations by Self-Organization: The LASSO model. Neurocomputing vol.6(3) pp 343-361 (1994)
17. Muruzabal, J.: On the Emulation of Kohonen's Self-Organization via Single-Map Metropolis-Hastings Algorithms. In ICCS (2001), pp 346-355
18. Osowski, S., Linh, T.H.: Fuzzy Clustering Neural Network for Classification of ECG Beats. In IJCNN (2000) pp 26-32
19. Płoński, P., Zaremba, K.: Improving Performance of Self-Organising Maps with Distance Metric Learning Method. In ICAISC(2012) pp 169-177
20. Sohn, S., Dagli, C.H.: Advantages of using fuzzy class memberships in self-organizing map and support vector machines. In IJCNN (2001), vol.3, pp 1886 - 1890
21. Song X., Hopke, P.K.: Kohonen neural network as a pattern recognition method based on the weight interpretation. Analytica Chimica Acta, vol.334, pp 57–66 (1996)
22. Wongravee, K., Lloyd, G.R., Silwood, C.J., Grootveld, M., Brereton, R.G.: Supervised Self Organizing Maps for Classification and Determination of Potentially Discriminatory Variables: Illustrated by Application to Nuclear Magnetic Resonance Metabolomic Profiling. Analytical Chemistry, vol.82, pp 628-638 (2010)