CONTROVERSIAL EMPIRICAL RESULTS ON BATCH VERSUS ONE PASS ONLINE ALGORITHMS

A. I. González, M. Graña[1]
Grupo de Inteligencia Computacional
Facultad de Informática, UPV/EHU
Apdo. 649, 20080 San Sebastián, España
ccpgrrom@si.ehu.es

Abstract *– We present controversial empirical results about the relative convergence of batch and online neural network vector quantization (VQ) learning algorithms. It is the commonplace belief that online algorithms have a very slow convergence, while batch versions are faster and give better results. However we propose an online realization of the Self-Organizing Map (SOM) and the Neural Gas algorithm, with learning parameters tuned to perform the training in a one-pass over the sample, that are faster than their batch counterparts and that improve them on some benchmark data sets. Although these results are too limited to claim any kind of superiority of the One Pass algorithms over their batch counterparts, they nevertheless must be taken into account as existence proofs contradicting the claim that the batch realizations are faster and more accurate than online algorithms.*

Keywords – SOM, Neural Gas, Batch, One pass online

1 Introduction

Vector Quantization [9, 10, 12] is a technique that maps a set of input vectors into a finite collection of predetermined codevectors. The set of all codevectors is called the *codebook*. In designing a vector quantizer, the goal is to construct a codebook for which the expected distortion of approximating any input vector by a codevector is minimized. This task has been tackled by with Competitive Neural Networks (CNN) [1]. Among them, the two architectures we focus on in this paper: Self Organising Map (SOM) [3, 14, 15] and the Neural Gas (NG) [17]. Both algorithms have the appearance of stochastic gradient descent (online) algorithms [8] in their original definitions, that is, whenever an input vector is presented, a learning (adaptation) step occurs. It has been shown that an online version of the NG algorithm can find better local solutions than the online SOM [17].

So called batch algorithms correspond to deterministic gradient descent algorithms. The adaptation is performed based on the whole data sample. As it is well known, one principal attractive of SOM is its capability for non-linear dimension reduction based on its topological preservation. Some works [3] have pointed its value for vector quantization as a robust initialization step for the fine

[1] The work is partially supported by MEC grants DPI2003-06972 and VIMS-2003-20088-c04-04, and UPV/EHU grant UE03A07.

tuning of the Simple Competitive Learning. The batch version of SOM was already proposed in [14] as a reasonable speed-up of the online SOM, with minor solution quality degradations. In the empirical analysis reported in [6], the main drawback for the batch SOM is their sensitivity to initial conditions, and the bad organization of the final class representatives, that may be due to poor topological preservation. Good execution instances of the batch SOM may improve the solutions given by the online SOM. On the other hand, the online SOM is robust against bad initializations and provides good topological ordering, if the adaptation schedule is smooth enough. If the goal of applying the SOM is Vector Quantization, then the topological preservation is not relevant, therefore the batch realizations may be of interest.

Neither the topological preservation nor the final ordering of the codevectors is relevant for Neural-Gas. In fact codevectors are updated according to their distance ranking to each input vector in the online case, amounting to perform the codebooks reordering for each input presentation. The batch version of the Neural-Gas algorithm has been studied in [18] as an algorithm for clustering data. It too has been proposed as a convenient speed-up of the online Neural Gas.

Both the online and batch algorithm versions imply the iteration over the whole sample several times. The approach that we propose is top visit only once the sample data: One Pass realizations. This adaptation framework is not very common in the neural networks literature; in fact, the only related reference that we have found is [4]. The effective scheduled sequences of the learning parameters applied to meet the fast adaptation requirement apparently fall far from the theoretical convergence conditions. However, as we shall see, in some cases the distortion results are competitive with the conventional SOM and Neural-Gas online and batch versions. The results presented in this paper must be interpreted as a kind of empirical existence proof of the assertion "One Pass realizations may perform equal or better than batch and online realizations" for the SOM and Neural-Gas. If we take into account the computation time, the One Pass realization improvement becomes spectacular. For independent verification, the Matlab code of the experiments described in this paper is available in the following web address:
`www.sc.ehu.es/acwgoaca/investigacion/proyectos/wsom05.`

Section 2 presents the formal definition of the algorithms. Section 3 gives the experimental results and section 4 is devoted to conclusions and discussion.

2 Algorithm definitions

Let it be $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$ the input data sample real valued vectors and $\mathbf{Y} = \{\mathbf{y}_1, .., \mathbf{y}_c\}$ the set of real valued codevectors (*codebook*). The design of the codebook is performed minimizing the error/distortion function *E*:

$$E = \sum_{i=1}^{n} \left\| \mathbf{x}_i - \mathbf{y}_{k(i)} \right\|^2; \quad k(i) = \underset{j=1,..,c}{\operatorname{argmin}} \left\{ \left\| \mathbf{x}_i - \mathbf{y}_j \right\|^2 \right\} \tag{1}$$

Each algorithm described below has some control parameters, like the learning ratio, the neighbourhood size and shape, or the temperature. The online realizations usually modify their values following each input data presentation and adaptation of the codebook. The batch realizations modify their values after each presentation of the whole input data sample. Both online and batch realizations imply that the input data set is presented several times. On the contrary, the One Pass realizations imply that each input data is presented at most once for adaptation, and that the control parameters are modified after each presentation.

2.1 One-pass version of Self-Organizing Map and Neural Gas

The SOM is a particular case of the general Competitive Neural Network algorithm:

$$\mathbf{y}_i(t+1) = \mathbf{y}_i(t) + \alpha(t)\,\mathrm{H}_i\left(\mathbf{x}(t), \mathbf{Y}(t)\right)\left(\mathbf{x}(t) - \mathbf{y}_i(t)\right) \tag{2}$$

Where t is the order of presentation of sample vectors. We denote by $\mathrm{H}_i(\mathbf{x}, \mathbf{Y})$ the so-called neighbouring function, and by $\alpha_i(t)$ the (local) learning rate. In the case of a conventional online realization, t corresponds to the iteration number over the sample, and the learning rate and neighbour value is fixed during iteration. In the case of One Pass realization, t corresponds to the input vector presentation number, and the learning rate and neighbour value is updated during iteration. In the experiments, the learning rate follows the expression [5]:

$$\alpha(t) = \alpha_0\left(\alpha_n / \alpha_0\right)^{\frac{t}{n}} \tag{3}$$

Where $\alpha_0$and α_n are the initial and final value of the learning rate, respectively. Therefore after n presentations the learning rate reaches its final value. In the case of the SOM, the neighbouring function is defined over the space of the neuron (codevector) indices. In our work, we assume a 1D topology of the codevector indices. The neighbourhoods considered decay exponentially following the expression:

$$\mathrm{H}_i(\mathbf{x}, \mathbf{Y}) = \begin{cases} 1 & |w - i| \le \left\lceil h_0 \left(h_n / h_0\right)^{8t/n} \right\rceil - 1; \quad w = \operatorname{argmin}\left\{\left\|\mathbf{x} - \mathbf{y}_k\right\|^2 k = 1,..,c\right\}, 1 \le i \le c \\ 0 & \textit{otherwise} \end{cases} \tag{4}$$

The initial and final neighbourhood radius is h_0 and h_n, respectively. The expression ensures that the neighbouring function reduces to the simple competitive case (null neighbourhood) after the presentation of the first $1/8$ inputs of the sample. With this neighbourhood reduction rate, we can get after a quickly initial ordering of codevectors, a slow local fine-tuning. We proposed this scheduling in **[11]** to approach real-time constraints and other authors have worked with this idea **[3]** in the context of conventional online realizations.

The Neural Gas introduced in **[17]** shares with the SOM the structure shown in equation (2) it is characterized by the following neighbouring function:

$$\mathrm{H}_i(\mathbf{x}, \mathbf{Y}) = \exp\left(-\mathit{ranking}(i, \mathbf{x}, \mathbf{Y}) / \lambda\right) \tag{5}$$

The *ranking* function returns the position $\{0,...,c-1\}$ of the codevector $\mathbf{y}_i$ in the set of codevectors ordered by their distances to the input **x**. All codevectors are updated, there are not properly defined neighbours, but the temperature parameter λ decays exponentially according to the following expression:

$$\lambda(t) = \lambda_0\left(\lambda_n / \lambda_0\right)^{\frac{t}{n}} \tag{6}$$

Where $\lambda_0$and λ_n are its initial and final value. The expression ensures that the neighbouring function reduces to the simple competitive case (null neighbourhood) as it happens with SOM.

In the case of his online version, t would correspond to the number presentation, and the temperature parameter value would be fixed for all the input samples during a complete presentation of the input data set.

2.2 Batch version of Self-Organizing Map and Neural Gas

Kononen's Batch Map [14, 15] defined the batch version of SOM algorithm. Among its advantages, there is no learning rate parameter and the computation is faster than the conventional online realization. This algorithm can be viewed as the LBG algorithm [16] plus a neighbouring function. When the input data set is completely presented, each input sample is classified in the Voronoi region defines by the winner codevector $y_w(t)$:

$$\forall i, 1 \le i \le c, \left\| x(t) - y_w(t) \right\| \le \left\| x(t) - y_i(t) \right\| \tag{7}$$

Where *t* corresponds to the iteration number over the sample and parameter values are fixed during iteration. To recalculate the centroid of regions an arithmetic mean is applied to a region and his neighbour regions as follow:

$$y_i(t) = \sum_{x(t) \in U_i} x(t) \Big/ n(U_i) \tag{8}$$

Where U_i is the union of Voronoi regions corresponding to the codevectors that lie up to a certain radius $h(t)$ from codevector *i*, in the topology of the codevector indices. And $n(U_i)$ means the number of samples x(*t*) that belong to U_i. To determine the radius of the neighbourhood we applied the following function:

$$h(t) = \left\lceil h_0 \left(h_n / h_0 \right)^{\frac{t}{n}} \right\rceil - 1 \tag{9}$$

The expression ensures that the neighbouring function reduces to h_n after *n* iterations. In the experiments, it takes value 0.1, which implies that its operation is equivalent to LBG or k-means: the centroid of a region is the arithmetic mean of the input sample that lies in this unique region.
In the Batch SOM, all neighbours have the same contribution to the centroid calculation, as the SOM online realization. A definition of Batch Neural-Gas arises from the thought of changing the contribution to the codebook in function of neighbour-region distances, imitating the online realization of Neural-Gas. We produce this effect applying a weighting mean as follows:

$$y_i(t) = \sum_{x(t)} x(t) w_{x(t)} \Big/ \sum w_{x(t)} \tag{10}$$

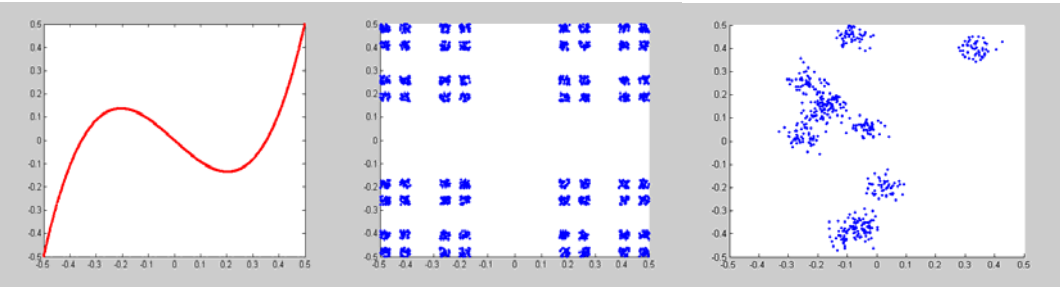
Where $w_{x(t)}$ is the weighting term for the input samples in the Voronoi region, defined by y_i codevector, given by expression:

$$w_{x(t)} = \exp\left(- ranking\left(i, \mathbf{x}(t), \mathbf{Y}(t) \right) \Big/ \lambda(t) \right) \tag{11}$$

The ranking function and temperature parameter λ are equal than in the one-pass case. The neighbour-region contribution decays exponentially due to the evolution of λ in equation (6). As with the Batch SOM, the Batch Neural-Gas converges to the LBG algorithm: only the region corresponding to the codebook contributes to its calculus.

3 Experimental results

In this paper, we have done the computational experiments on three 2D benchmark data sets which have already been used in [5,7] for evaluation of clustering and VQ algorithms. They are visualized in figure 1: (a) S-shaped distribution (b) Three-level Cantor distribution, (c) A mixture of Gaussian distributions.



(a) (b) (c)

Figure 1: The three benchmark data sets (a) S-shaped, (b) Cantor set and (c) mixture of Gaussian distributions

The first distribution is constructed with 2000 data points that fall on an S-shaped curve defined by the equation $y = 8x^3 - x$, where x is uniformly distributed in the interval [-0.5,0.5]. The three-level Cantor set (2048 data points) is uniformly distributed on a fractal; it is constructed by starting with a unit interval, removing the middle third, and then recursively repeating the procedure on the two portions of the interval that are left. And the third data set is a collection of 500 data points generated by a mixture of ten Gaussian distributions with $\bar{x} \in [-0.5, 0.5]^2$ and $\sigma^2 = 0.001$. The three sets are shown in figure 1.

The codebook initialization used in this paper and reflected in the results of figure 2 is a random selection of input sample data. The codebook size is set to $c = 16$ codevectors. The maximum number of sample presentations has been established at $n = 50$ for conventional online and batch realizations of the algorithms. Nevertheless, we introduce a stopping criterion on the relative decrement of the distortion; the process will stop if it is not greater than $\xi = 0.001$. For SOM algorithms the neighbourhood initial and final parameter values have been set to: $h_0 = c/2 + 1$; $h_n = 0.1$, and for NG algorithms they have been set to $\lambda_0 = c/2$; $\lambda_n = 0.01$. In both one-pass version algorithms the learning rate values are $\alpha_0 = 0.5$ and $\alpha_0 = 0.005$. We have executed 100 times each algorithm. In figures 2a, b, c we present the mean and 0.99 confidence interval of the distortion results of the tested algorithms for each data set. We have also taken into account the computation time. In figures 2d, e, f the y-axis corresponds to the product of the final distortion and the computation time as measured by Matlab. The algorithms tested are: the online conventional realizations of SOM and Neural Gas (NG), the batch versions (BSOM and BNG) and the online One Pass realizations (SOMOP and NGOP). The inspection of the figure reveals that the relative efficiencies of the algorithms measured by the final distortion depend on the nature of the data. For example, the One Pass SOM improves the online and batch algorithms on the Cantor dataset; it is similar on the S-shaped data set and falls behind in the Gauss data set. Although this is not the main concern of this paper, the distortion results show that the Neural Gas improves the SOM most of the times, confirming the results in the literature [17]. The batch realization sometimes improves the online realization (Cantor and S-shape), sometimes not (Gaussian).

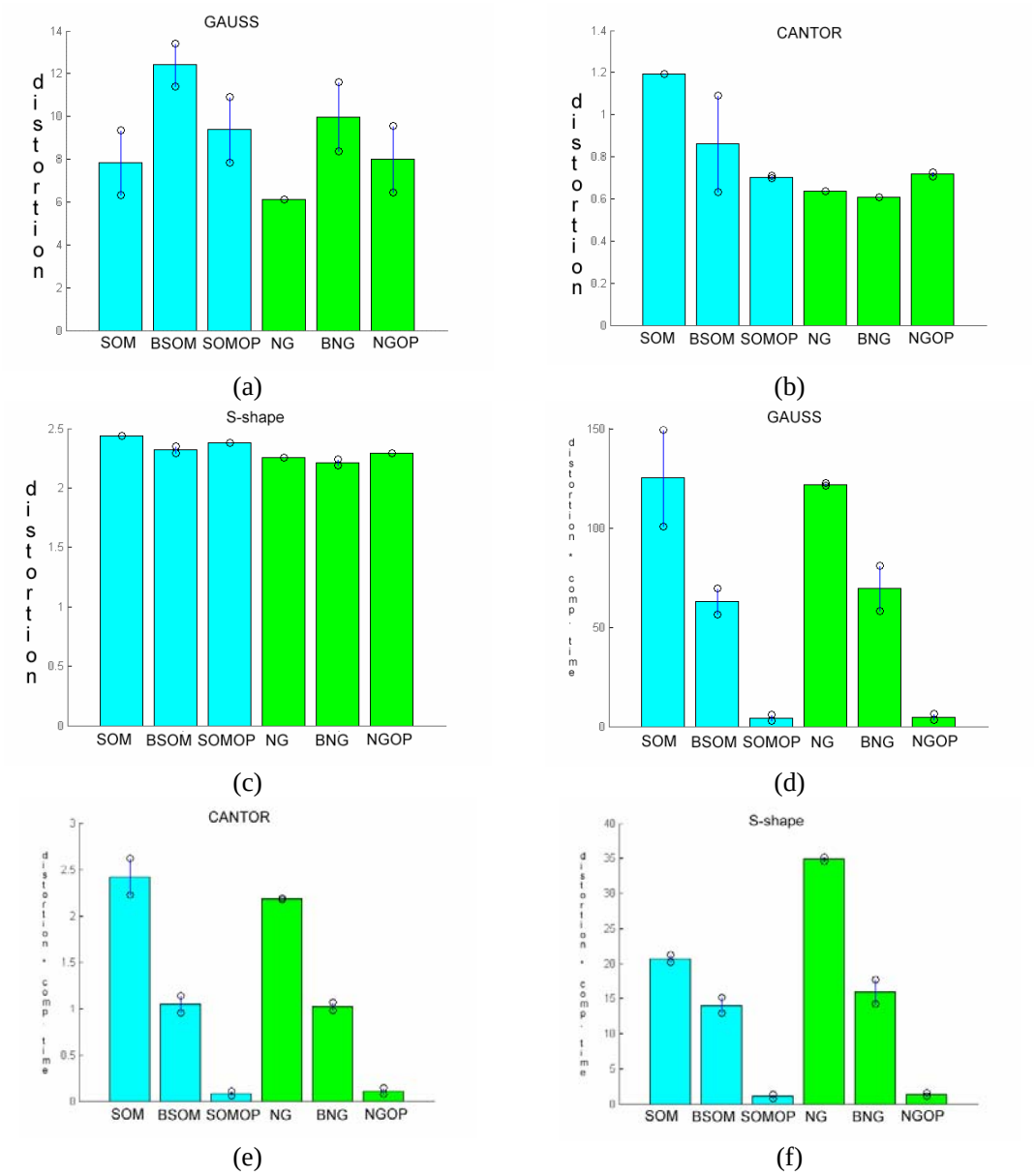


Figure 2. Results on the three benchmark algorithms. The distortion on the (a) Gaussian data, (b) Cantor data set and (c) S-shaped data. The distortion times the computational time for (d) Gaussian data, (e) cantor data, and (f) S-shaped data.

The main motivation of our work was to compare batch and One Pass realizations, from figures 2a, 2b and 2c the One Pass SOM improves the batch SOM in some cases (Gaussian and Cantor) and is almost equal in the S-shape data, when all the algorithms behave almost identically. However, the One Pass Neural Gas improves the batch realization only on the Gauss data set.

When we take into account the computation time in the plots of figures 2d, 2e and 2f, the improvement of the One Pass realization over the batch and conventional realizations is spectacular. It can also be appreciated the improvement of the batch realization over the conventional online realization. We have used the product of time and distortion instead of the ratio distortion/time in order to maintain the qualitative interpretation of the plots, because with the latter ratio greater would be better, contrary to distortion results. The conclusion from figures 2d, 2e and 2f is that One Pass realizations may be worth for time critical applications even in the case (not appearing in this paper) that the batch or online realizations give much better distortion results.

4 Discussion and Conclusions

As it is pointed out in [13], besides its statistical interpretation as stochastic gradient descent algorithms, robust Vector Quantization algorithms, a category where both SOM and Neural Gas fall, can be interpreted and analyzed in the framework of Continuation Theory [2]. The key idea in continuation methods is to optimize an objective function E by tracking solutions of a family of objective functions E_P in the limit of $P \to 0$ where $E_P = E$. These procedures primarily try to avoid undesired local minima by proposing a family of functions E_P increasingly smooth with P. In the case of SOM and Neural Gas the limit objective function is the Vector Quantization distortion of equation (1). Therefore, the whole learning process is oriented towards the minimization of this function, and the successive values of the control parameters correspond to a succession of objective functions. Under this interpretation, the conditions in equation (2) are less significant to obtain good convergence properties. However, this interpretation is only of application to problems where the minimization of the distortion in equation (1) is the goal. It does not apply to problems where dimension reduction and topological preservation are the goals.

We do not claim in this paper any optimality of the learning control parameters schedules for the One Pass algorithms. We think that the formal derivation of optimal schedules or the proof of the convergence discussion above is outside the scope of the paper. Also, the results are restricted to quantification distortion, therefore any discussion involving supervised training algorithms or topological preservation is outside the scope of the paper and, in our humble opinion, a confusion of terms and computational paradigms.

It can be argued that our results are of no use if they do not apply to high dimensional data. Let us recall that the results presented here are a kind of counterexample proof against the assumed superiority of batch algorithms. From our point of view the reasoning we try to rise is: “if the batch algorithms do not improve significantly over One Pass realizations in some toy examples, how can be asserted that they would be superior in real life high dimensional examples?” The sensible reasoning is that the curse of dimensionality will affect both approaches and, therefore, both of them will degrade as the data dimensionality increases. We do not know of any formal work showing that batch SOM or NG algorithms improve their relative efficiency as the data dimensionality increases.

In this paper we have presented empirical results showing that online One Pass training give competitive distortion results, and may even improve, than conventional multiple pass online training and batch training. If we take into account the computation time, the performance improvement of the One Pass training may be spectacular. We invite the reader to visit the Internet address given in the introduction and to download the data and the Matlab code to verify our assertions. Additional results, like the visualization of the final codebooks and the topological ordering of the SOM solution, or the results under diverse initialization schemas can be found there, or can be computed with the code provided. Further work will be addressed to extend the computational experiences to other data sets.

References

[1] S.C. Ahalt, A.K. Krishnamurthy, P. Chen, D.E. Melton (1990), Competitive Learning Algorithms for Vector Quantization, *Neural Networks*, vol. 3, p.277-290.

[2] E. L. Allgower and K. Georg, Numerical Continuation Methods. An Introduction, Vol. 13, Springer Series in Computational Mathematics. Berlin/Heidelberg, Germany: Springer-Verlag, 1990.

[3] E. Bodt, M. Cottrell, P. Letremy, M. Verleysen (2004), On the use of self-organizing maps to accelerate vector quantization, *Neurocomputing*, vol 56, p. 187-203.

[4] C. Chan, M. Vetterli (1995), Lossy Compression of Individual Signals Based on String Matching and One Pass Codebook Design, *ICASSP'95*, Detroit, MI.

[5] C. Chinrungrueng, C. Séquin (1995), Optimal Adaptive K-Means Algorithm with Dynamic Adjustment of Learning Rate, *IEEE Trans. on Neural Networks*, vol. 6(1), p.157-169.

[6] Fort J.C., Letrémy P., Cottrell M. (2002), Advantages and Drawbacks of the Batch Kohonen Algorithm, , in M. Verleysen (ed), Proc. of ESANN'2002,Brugge, Editions D Facto, Bruxelles, p. 223-230.

[7] B. Fritzke (1997), The LBG-U method for vector quantization - an improvement over LBG inspired from neural networks, *Neural Processing Letters*, vol. 5(1) p. 35-45.

[8] K. Fukunaga (1990), *Statistical Pattern Recognition*, Academic Press.

[9] A. Gersho (1982), On the structure of vector quantizers, *IEEE Trans. Inf. Th.*, 28(2), p.157-166.

[10] A. Gersho, R.M. Gray (1992), *Vector Quantization and signal compression*, Kluwer.

[11] A. I. Gonzalez, M. Graña, A. d'Anjou, F.X. Albizuri (1997), A near real-time evolutive strategy for adaptive Color Quantization of image sequences, *Joint Conference Information Sciences*, vol. 1, p. 69-72.

[12] R.M. Gray (1984), Vector Quantization, *IEEE ASSP*, vol. 1, p.4-29.

[13] T. Hofmann and J. M. Buhmann (1998) Competitive Learning Algorithms for Robust Vector Quantization *IEEE Trans. Signal Processing* 46(6): 1665-1675

[14] T. Kohonen (1984) (1988 2nd ed.), *Self-Organization and associative memory*, Springer Verlag.

[15] T. Kohonen (1998), The self-organising map, *Neurocomputing*, vol 21, p. 1-6.

[16] Y. Linde, A. Buzo, R.M. Gray (1980), An algorithm for vector quantizer design, *IEEE Trans. Comm.*, 28, p.84-95.

[17] T. Martinetz, S. Berkovich, K. Schulten (1993), Neural-Gas network for vector quantization and his application to time series prediction, *IEEE trans. Neural Networks*, vol. 4(4), p.558-569.

[18] S. Zhong, J. Ghosh (2003) A Unified Framework for Model-based Clustering *Journal of Machine Learning Research* 4:1001-1037