Visualization techniques of self-organizing maps

Frederico Fernandes
Instituto Superior Técnico
frederico.fernandes@tagus.ist.utl.pt

Abstract: Neural networks try, in a computing way, to simulate human brain, including its behavior, by making errors and learning and thereby making new discovers. Self-organizing maps are part of a neural network group based on competitive networks where competition is used as a way of learning. They try to find similarities between data, based only on input data, grouping similar data to each other and thereby forming clusters. Self-organizing maps learning it's unsupervised. It can adapt its behavior without any previous knowledge and also, without human intervention. The maps can make connections between the observations that were made and the expected result. Its result enables improvements in future decisions. The main point of this master thesis is the self-organizing maps speed improvement and a presentation of a possible way to visualize large dimensions maps, once the only way to do it, it's by getting to know if the learning process was achieved.

Keywords: Kohonen maps, self-organizing maps, neural networks, clusters, data mining, visualization, learning.

1. Introduction

Nowadays neural network models are used as alternative to the traditional models of cluster, due to their characteristics of performance, facing incomplete data and, to their capacity to establish relations between them. [1]
There are two main groups in clustering techniques: the group of the supervised and unsupervised. The supervised consider that there is a prior knowledge of the information, while unsupervised divide classes, without requiring such knowledge. The unsupervised learning observes inputs and arranges their behavior without such needing knowledge. [1]
The unsupervised learning observes the inputs and adapts its behavior, without the needing to indicate a way to make associations between the observations and the expected result. Frequently, it is produced, as a result, a new representation of the observed data, allowing improvements in future decisions and responses.
In 1982, the self-organizing maps were introduced by Teuvo Kohonen [2] and are part of a class of neural networks, whose learning is unsupervised and its main goal is to reduce data's dimensions, using a self-organized network. The applications of self-organizing maps extend to several areas such as research on the internet [3], bioinformatics [4], finance [5] and its increasing importance.
In this methodology, the main feature is the use of neurons that are competing to see which one generates more alternative outputs. Basically, the result of self-organizing maps is a two-dimensional mesh, where we can view the most active neurons and understand the relationships between variables.
The main difference between the self-organizing maps method and the conventional statistical methods, such as k-means, logistic regression (supervised method) is the fact that, from the last one you won't be able make a two-dimensional output. [6]

2. Related Work

2.1. Clustering

Clustering is a technique that allows us to group a set of objects, physical or abstract, according to their similarities, which is also used in data mining. [7,8] The main objective is to reduce the data size, therefore, it may be considered as a type of compression [9]. A cluster is a set of objects that are similar to each other and different from the others in other clusters. We can distinguish and classify each set of objects. When we have large data sets the method takes a lot of time, so it's preferable to choose the inverse method: making the partition of data set, according to their similarities (for example, using clustering) and then labeling and classifying each group.
We can divide the clustering methods in two types: hierarchical and particional. In the first one, each generated cluster can be understood as a deviation from the original one, while in the second type is made a separation by proximity to the centroid of each cluster.
Cluster analysis concept, was used for the first time by Tyron [10], and consists in several algorithms and methods for grouping objects of the same type in their respective categories, ie, is analysis tool that makes the distribution of different objects groups, so that the similarity between two objects is maximal if they are in the same group and minimal otherwise.
Cluster analysis is a set of techniques where the main objective is to group objects according to their characteristics, forming homogeneous groups or clusters. [11] In each cluster, the objects tend to be

identical to each other but different from the objects that are in other cluster. The obtained clusters show an internal consistence (within each cluster), as well as an external heterogeneity (between clusters). In case of a successful clustering, when we're representing the objects on a graph, the ones that are inside of the clusters will be very close to each other while different clusters will be distant.
Cluster techniques are advantageous since they do not require knowledge from data to be processed, however, when applying, they have two disadvantages: Before you begin the clustering process is necessary to define the number of clusters that we want to form. Different number of clusters gives us different results. The second disadvantage is the need of the user knowledge to conclude clustering operations results. The most widely known and more used methods to partition are the algorithms k-means, k-medoids and their variations.

2.2. Self-Organizing Maps

Frequently the self-organizing maps (SOM) create a bidimensional map. Neighborhood function is used by self-organizing maps, and makes the preservation of all topological properties of the input space possible. Thereby it distinguishes them from all other neural networks. They are very useful to visualize high-dimensional data in low dimensions.
Self-organizing maps have two phases of traning. The first one is called training phase, and second one is mapping. In a SOM, the training phase it is done to find a best matching unit (BMU). This unit is the one, that has the smallest Euclidean distance between a weight vector and a input space vector. Then, and assuming the best matching unit as center, we search for neighboors in a distance obtnained by neighboorhood function. While network training, this distance is reduced by a function, that only catches the closer neurons to the best matching unit. All neurons that are in that neighboorhood are updated at the next phase iteration. The neurons weights are updated, proportional to the learning rate and to the distance that they are from best matching unit. In other words, during the learning each neuron inside the neighboorhood of winner neuron participates in learning. The steps are:

- Initialize randomly the weight vectors.
- Apply that weights to a input vector and determine the best matching unit.

If x is an input vector, the winner neuron would be:

$$\|x - m_c\| = \min_i \|x - m_i\| \quad (1)$$

Where c is refering to winner neuron. Then we update all the winner neuron weights and all other neurons that are in the neighboorhood, using the function:

$$\begin{aligned} m_i(t_k + 1) &= \\ &= m_i(t_k) + \alpha(t_k) \cdot (x(t_k) - m_i(t_k)) \end{aligned} \quad (2)$$

3. Proposed Model

Previously it was shown that the self-organizing maps are a data visualization technique that allows the reduction of dimensions number by using a self-organizing neural network.
There is a problem in visualizing multidimensional data due to the lack of human capacity to observe more than three dimensions. Concerning that, the proposed model allows us to view and understand these data. By using a map of one or two dimensions that groups the similar data, clustering them, we reduce the number of dimensions.
Through this model we will be able to solve the problem that comes when we try to visualize weigh vectors with more than three dimensions or, more specifically, when the input space have vectors with more than three dimensions.
The visualization is necessary to accurate if a map is properly developed once it is essential to certify that it does not have defects in its visual representation.

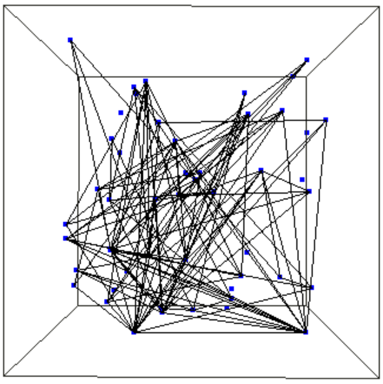

Figure 1 – Input space with three dimensions.

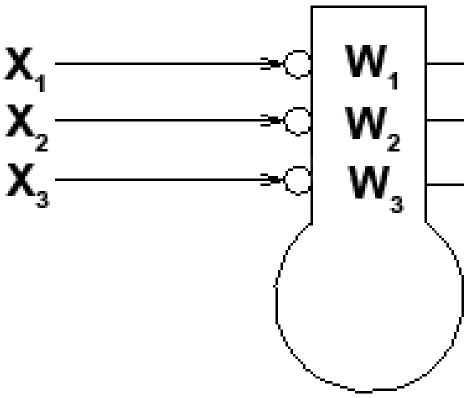

Figure 2 – Three dimensional weight vector.

In Figure 1 a tridimensional example of an input space its reproduced and as we can conclude, when-

ever the dimensions are more than three it is impossible to see the representation due to the incapacity of projecting points on a space beyond the tridimensional. In Figure 2 there is weigh vector with three dimensions where each point in the Figure 1 is represented by a weigh vector.
To solve this problem it is proposed the use of the mean value to reduce the number of the input space dimensions, and make it possible to view the number of dimensions. The model has two different stages. The first will be called "phase A", where the self-organizing network learns with, only, and two dimensions. For example, when the weigh vectors have 16 dimensions we do the mean value in order to get two dimensions. As it is shown:

$$X_1 = \left(\frac{x_1 + \cdots + x_8}{8}\right) \qquad X_2 = \left(\frac{x_9 + \cdots + x_{16}}{8}\right) \quad (3)$$

For evenly distributed data and a Euclidean distance function we use the mean value where the best mapping function which corresponds to the computation of the mean value of the projected point (projection on the bisecting line) is the orthogonal projection.
To support this statement, we know that for a given vector $\vec{x}$ of a dimension m and determining the vector $\vec{a} = (a_1, a_2, \dots, a_k, \dots, a_m)$ with $a_1 = a_2 = \cdots = a_m = \alpha$ according to the Euclidean distance function. As we can see, each component it's equal in the vector $\vec{a}$. How can we chose the value of α? How to minimize the distance of $d(\vec{x}, \vec{a})$:

$$\min_{\alpha}\left(\sqrt{(x_1 - \alpha)^2 + (x_2 - \alpha)^2 + \cdots + (x_m - \alpha)^2}\right) \quad (4)$$

$$0 = \frac{\partial d(\vec{x}, \vec{a})}{\partial \alpha} = \frac{m \cdot \alpha - (\sum_{i=1}^{m} x_i)}{\sqrt{m \cdot \alpha^2 + \sum_{i=1}^{m} {x_i}^2 - 2 \cdot \alpha \cdot \sum_{i=1}^{m} x_i}} \quad (5)$$

So the solution is:

$$\alpha = \frac{\sum_{i=1}^{m} x_i}{m} \quad (6)$$

Which is the mean value or the mean value of the vector $\vec{x}$ and, the reduced space of two dimensions is [12, 13 and 14]:

$$\sqrt{\frac{m}{2}} \cdot d^{(2)}(x, y) \approx d^{(m)}(x, y) \quad (7)$$

In the Figure 3 we can see a graphic representation of the proposed model. It shows how it is possible, using this method, to achieve the equivalent of a weigh vectors visualization with more dimensions that the ones that are possible to see with a human eye.

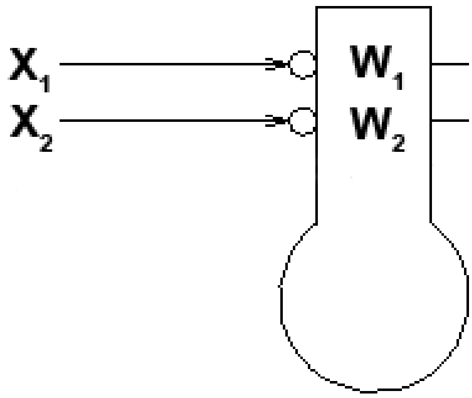

Figure 3 – Weight vector of equation.

In the second stage, called "phase B", the weigh vectors are replaced by others whose dimensions are equal to the originals, where each half it's replaced by the mean value. From the example used in "phase A" we can see in Figure 4 (a) the vectors that were used on first stage, and in the Figure 4 (b) we have the weigh vectors that will be used to the "phase B" of the learning procedure.

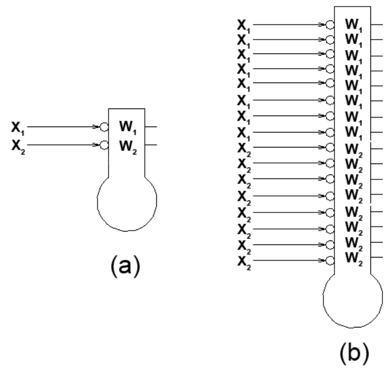

Figura 4 – (a) Weight vector of na equation, (b) Weight vector after expanding to "phase B".

Looking to the Figure 5 it gets easier to understand the differences between the two methods. The Kohonen algorithm uses the same dimensions in the first stage, the learning stage, and in the second stage, the labeling stage. The proposed model reduces the weigh vectors dimension in the first stage, "phase A", making it possible to be visually represented and then, doing its expansion to the second stage, "phase B", using the mean values scaled to the original dimensions.
In the second stage of the proposed model, since it has the original dimensions it wouldn't be possible to view the result and, concerning the previous example, we would have 16 dimensions where the first half it's composed by x_1 and the second half it is composed by x_2.

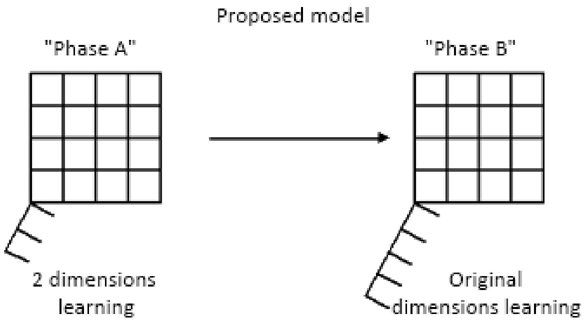

Figure 5 – Proposed model explanation.

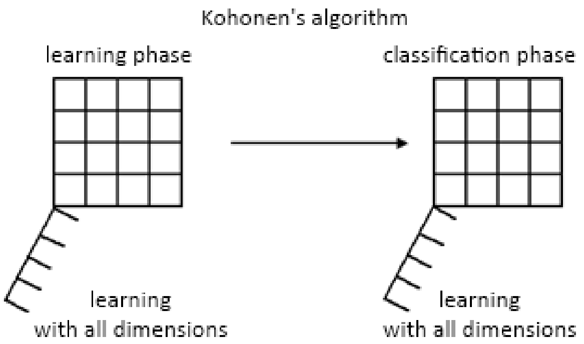

Figure 6 – Kohonen algorithm.

In order to make it possible to be visualized, the mean value of half the total dimensions is done, leaving only two dimensions, making it possible to be represented and to be visualized.
We can obtain three interesting results using the proposed model. The first is the visualization of data with more than three dimensions; the second is the optimization of Kohonen's algorithm, and last but not least, the network arrangement is more noticeable.

3.1. Implementation

To implement this model its used java programing language that allows portability between different systems. It was created an applet so that web access would be possible and then reach the largest number of users.
The program allows us to test the proposed model for all the files in the next chapter of this master thesis, and all of its parameters can be personalized. The most important parameters are the time spent on each stage and the moment of the transition of "stage A" to "stage b" occurs. There are also many other customizable parameters in Table 2 of the next chapter. The original Kohonen algorithm test is admitted in these files, and all of the previous parameters can also be modified, except in the transition stage where it is not applicable.

4. Experiments

This chapter presents test results performed after the implementation of the proposed model. To conduct these experiments three types of data files were used. The different experiments coverage was higher, so the results were more conclusive. Due to the paper size, it was decided to not include all the experiments, but only a few[1].
The first two files used are a high dimensional voice recognition representation and the last one composed by local scale-invariant feature transform (SIFT) descriptors. Each pattern corresponds to a high dimensional vector. For each file type multiple experiments were made, by using the Kohonen and proposed model algorithm. To perform the learning experiments for both algorithms, we used standard parameters, to make them able to be compared.
In the developed application the procedure of learning using Kohonen algorithm is done by the implementation of a decreasing time function. Approximate function can be seen in Figure 8.

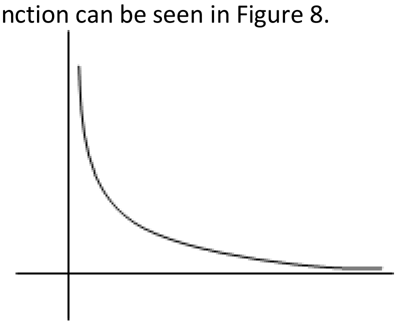
Figure 7 – Original function.

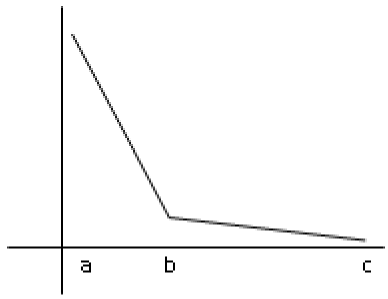

Figure 8 - Approximate function used in the proposed model.

Ft0, Ft1, and Ftmax correspond to the values of approximation function and respectively to the points in Figure 8, a, b and c. Ht0 is the value where the function has a point, and decreases to point b, where there is the value of Ht1. Resembling, the function at the point c will have the value of Htmax and will decrease from Ht1 to Htmax, from b to c. Empirical experiments values can be seen in the following tables.
In Table 2 there is a column called transition phase which is used as an input to the application to make it possible to indicate when the transition from the learning to the classification stage should be made.

[1] It’s possible to access all experiments in: http://web.tagus.ist.utl.pt/~frederico.fernandes/thesis/

Using the first row of Table 2 as example, it is clear that the maximum time of learning is 5000, the learning stage time, which is where the algorithm will learn in two dimensions, is 1000 units and the classification phase is 4000. Summarizing, the "phase A" of the proposed model is executed during 1000 units of time and the "phase B" at the 4000.

In each one of the file types three different dimensions of Kohonen maps were used, to obtain more explicit results and to get valid conclusions. The dimensions used may vary depending on the number of neurons that we want for each map, maps with 100, 225, and 400 neurons were used which make it possible for the Kohonen maps to have dimensions of 10x10, 15x15, and 20x20, respectively. Tables 1 and 2 were both used for each dimension in each map, to make higher experiments coverage.

Step	Tmax	Transition Phase	Ft0	Ht0	Ft1	Ht1	Ftmax	Htmax
40	5000		10	0,9	3	0,02	0	0,001
40	10000		10	0,9	3	0,02	0	0,001
40	20000		10	0,9	3	0,02	0	0,001

Table 1 – Learning values using Kohonen's algorithm.

Step	Tmax	Transition Phase	Ft0	Ht0	Ft1	Ht1	Ftmax	Htmax
40	5000	1000	10	0,9	3	0,02	0	0,001
40	5000	3000	10	0,9	3	0,02	0	0,001
40	5000	4000	10	0,9	3	0,02	0	0,001
40	10000	2000	10	0,9	3	0,02	0	0,001
40	10000	6000	10	0,9	3	0,02	0	0,001
40	10000	8000	10	0,9	3	0,02	0	0,001
40	20000	4000	10	0,9	3	0,02	0	0,001
40	20000	12000	10	0,9	3	0,02	0	0,001
40	20000	16000	10	0,9	3	0,02	0	0,001

Table 2 – Learning values using proposed model.

All images experiments are not included in the chapters below, but only a very few of them. Instead, it will be included all experiments result tables so it would be possible to get conclutions of them.

4.1. First file type

This first file type used was words vocal representation through four different speakers starting from nine to zero. Words are represented by 19 frequencies and are normalized to 26 time steps. There are 100 vectors and each one has 494 dimensions (19x26). The experiments used show the learning process of this file type using a network size of 20x20.

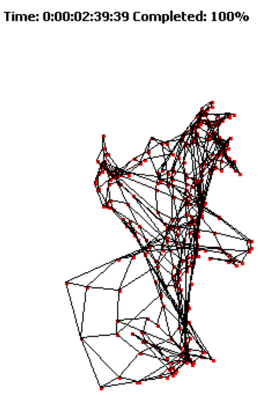

Figure 9 – Last iteration of learning process using Kohonen's algorithm in a network of 20x20 and with a tmax of 20000. The network did not learn properly.

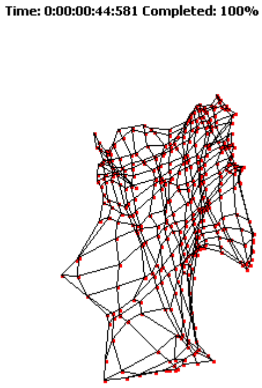

Figure 10 – Last iteration of learning process using Kohonen's algorithm in a network of 20x20 and with a tmax of 20000. The transition phase was 16000 and the network learnt properly.

The learning using Kohonen's algorithm was not good, because the network have a lots of defects. Its possible to see a time improvement in learning phase, using the algorithm of proposed model, and the network using this model could develop in a good way and presenting to us a good learn process. As we can see in Table 3, using a bigger transition phase makes the organization improvement in self-organization maps faster than using the original Kohonen's algorithm.

File	Algorithm	Tmax	Transition Phase	Time (minutes)
1	Kohonen	5000		00:29:969
2	Kohonen	10000		01:04:870
Figure 9	Kohonen	20000		02:39:390
3	Proposed Model	5000	1000	00:30:993
4	Proposed Model	5000	4000	00:11:963
5	Proposed Model	10000	2000	00:50:481
6	Proposed Model	10000	8000	00:23:447
7	Proposed Model	20000	4000	01:38:184
Figure 10	Proposed Model	20000	16000	00:44:581

Table 3 – Time results of a network with 20x20 dimensions using both algorithms for first file type. Note: the numbered files (1-7) are not included in this paper.

File	Algorithm	Tmax	Transition Phase	Time (minutes)
1	Kohonen	5000		00:23:409
2	Kohonen	10000		00:43:191
Figure 11	Kohonen	20000		01:22:642
3	Proposed Model	5000	1000	00:20:785
4	Proposed Model	5000	4000	00:14:284
5	Proposed Model	10000	2000	00:38:536
6	Proposed Model	10000	8000	00:25:383
7	Proposed Model	20000	4000	01:25:257
Figure 12	Proposed Model	20000	16000	00:54:319

Table 4 – Time results of a network with 20x20 dimensions using both algorithms for second file type. Note: the numbered files (1-7) are not included in this paper.

File	Algorithm	Tmax	Transition Phase	Time (minutes)
1	Kohonen	5000		00:09:397
2	Kohonen	10000		00:17:109
Figure 13	Kohonen	20000		00:32:306
3	Proposed Model	5000	1000	00:09:421
4	Proposed Model	5000	4000	00:08:790
5	Proposed Model	10000	2000	00:16:585
6	Proposed Model	10000	8000	00:15:410
7	Proposed Model	20000	4000	00:31:268
Figure 14	Proposed Model	20000	16000	00:28:131

Table 5 – Time results of a network with 20x20 dimensions using both algorithms for third file type. Note: the numbered files (1-7) are not included in this paper.

4.2. Second file type

The second file type used words vocal representation through one speakers (American English) starting from nine to zero. It was composed by 320 patterns and each voice pattern was represented by a binary matrix of dimension 16x16, corresponding to 16 frequencies in 16 time steps. If amplitude of a band k+1 is bigger than that one of band k, it is set to one, otherwise it is zero. There are 320 vectors and each one has 256 dimensions (16x16).
Assuming the experiments of last file type, for this file type was used the same experiments, so that the same consistency could be added to this master thesis. It was used a network with a size of 20x20 to run the experiments.

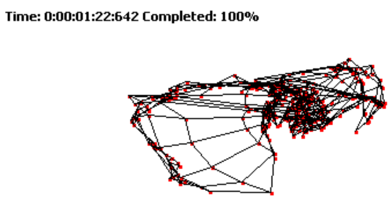

Figure 11 – Last iteration of learning process using Kohonen's algorithm in a network of 20x20 and with a tmax of 20000. The network did not learn properly.

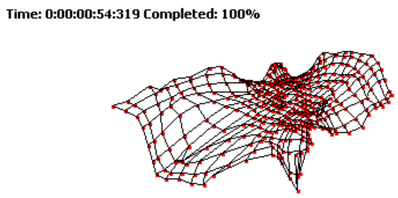

Figure 12 – Last iteration of learning process using Kohonen's algorithm in a network of 20x20 and with a tmax of 20000. The transition phase was 16000 and the network learnt properly.

In Figure 11, using Kohonen's algorithm, its not possible to visualize all the clusters and the network has a lot of defects. As concluded in file type number one, we had a big improvement in learning time, because by using the proposed model, the network learnt faster and with a better visualization of clusters. In Table 4, it's also possible to see, that the experimental results obtained for all the other parameters used, go according to the same conclusion.

4.3. Third file type

This last file type are composed by local scale-invariant feature transform (SIFT) descriptors[2]. Vectors of SIFT are built as concatenated orientation histograms of images. This file has 10000 vectors and each vector has 128 dimensions.
Although this file was different from the other two types of files, here was used the equivalent learning maps that were used before, so the network has the

[2] Files had been downloaded from: http://corpus-texmex.irisa.fr/

size of 20x20. The next figure contains the learning process using the Kohonen's algorithm.

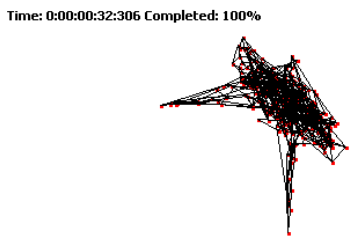

Figure 13 – Last iteration of learning process using Kohonen's algorithm in a network of 20x20 and with a tmax of 20000. The network did not learn properly.

Its very difficult to understand Figure 13 results, because all the clusters are too close together, so the network could not learn properly. Figure 14 shows learning development by using the proposed model.

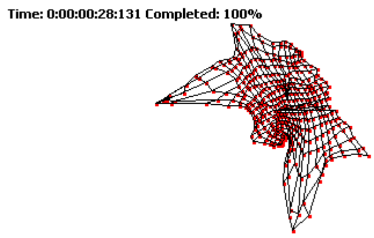

Figure 14 – Last iteration of learning process using Kohonen's algorithm in a network of 20x20 and with a tmax of 20000. The transition phase was 16000 and the network learnt properly.

As we can see in Figure 14, the result was very satisfactory, and the execution time was drastically lower. All the clusters are separated from each other, what makes them realize that the network has learnt in a proper way. Looking at Table 5 we could still conclude the same results of the two other files. Proposed model will improve the Kohonen's algorithm speed of learning and it will make a better and possible visualization.

4.4. Comments

After analyzing all the experimental tests, it is clear that the proposed algorithm is always the faster execution. This effect is mainly due to reduced dimensions the first stage of learning, ie, the "phase A" of the algorithm. The learning phase of Kohonen's algorithm is made with the overall dimensions, which in cases where these are very high, eg, 494 dimensions, the computer calculation made is much smaller, when these are reduced to just two making remarkable timing in its execution. Another important reason for the algorithm speed is slope of the learning line is minor, thereby increasing its performance.

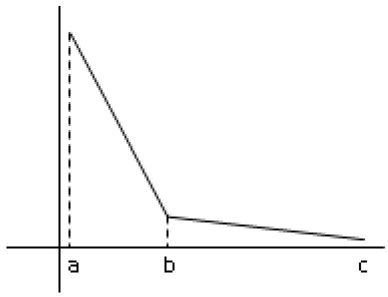

Figure 15 – Approximate function used in the proposed model.

In the proposed model algorithm of this paper the point b of the Figure 15 is controllable. That point refers to the transition phase value and, can be parameterized, making the slope of the line increase or decrease. When the slope of the line is smaller, in other words, when the Figure 15 point is farther from the point b, the mesh tends to get better learning results.

5. Conclusion

The use of self-organizing maps have a big importance in the large dimension data visualization, however, visualizing that data wouldn't be possible before this work. The only way to accurate if a Kohonen map has learnt correctly is by observing and checking if there aren't any defects. The learning time in multiple vectors with many dimensions is very long and it takes a great computational capacity.

To solve this problem we chose to use the mean value of the weights [12, 13 and 14] on the conversion to two dimensions, on which part of learning is developed with clear gains in efficiency of the overall process.

The empirical experiments and the visual inspection of the maps show that the proposed method speeds up the computation of the maps. They also prove that their learning is better done and make it possible, for us, to get better results than by using Kohonen's original algorithm.

5.1. Future Work

In the development of this work a fully configurable software was designed to study the learning process of self-organizing maps.

This software allows the use of the original Kohonen's algorithm and the proposed model, with clear gains in performance and in visualization.
With its use, certainly others who continue this work will have less dificulties.

6. Bibliography

1. Haykin, S. H.: Neural Networks – A Comprehensive Foundation, Prentice Hall, (1999)
2. Kohonen, T.: Clustering, Taxonomy, and Topological Maps of Patterns, Proc. 6th Int. Conf. On Pattern Recognition, October, (1982)
3. Kohonen, T., Kaski, S., Lagus, K., Salojärvi, J., Honkela, J. and V. Paatero et al.: Self-Organization Of a Massive Text Document Collection, IEEE Transactions on Neural Networks 11 (3) (2000), pp. 574-585, (2000)
4. Abe, T., Kanaya, S., Kinouchi, M., Ichiba, Y., Kozuki, T. and Ikemura, T.: Informatics for Unveiling Hidden Genome Signatures, Genome Research 13 (4) (2003), pp. 693-702, (2003)
5. Deboeck, G. and Kohonen, T.: Visual Explorations in Finance with Self-Organizing Maps, Springer, London , (1998)
6. Pao, Y. H.: Adaptative Pattern Recognition and Neural Networks, Addison-Wesley, (1989)
7. Cios, K. J. and Kurgan, L.: Trends in Data Mining and Knowledge Discovery, In: Pal, N.R. et al. (Eds.), Knowledge Discovery in Advanced Information Systems, Springer, (2002)
8. Murtagh, F.: Clustering Massive Data Sets, in J. Abello et al. (Eds.), Handbook of Massive Data Sets, Kluwer, (2000)
9. Boutros, P.C. and Okey, A.B.: Unsupervised Pattern Recognition: An Introduction to the Whys and Wherefores of Clustering Microarray Data. Brief. Bioinformatics 6, pp.331-343, (2005)
10. Tryon, R. C.: Cluster Analysis, Edwards Brothers, Ann Arbor, Michigan, (1939)
11. Hair, Joseph et al.: Multivariate Data Analysis, New Jersey: Prentice Hall, (1998).
12. Wichert, A.: Content-based Image Retrieval by Hierarchical Linear Subspace Method, Journal of Intelligent Information Systems, 31 (1): (pp. 85-107), (2008)
13. Wichert, A., Teixeira, P., Santos, P., Galhardas, H.: Subspace Tree: High Dimensional Multimedia Indexing with Logarithmic Temporal Complexity, Journal Of Intelligent Information Systems, 35 (3): (pp. 495-516), (2010)
14. Wichert, A., Verissimo, A.: CBIR with a Subspace-Tree: Principal Component Analysis Versus Averaging, Multimedia Systems, (2011)