ESANN'2005 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 27-29 April 2005, d-side publi., ISBN 2-930307-05-6.

A new learning algorithm for incremental self-organizing maps

Yann Prudent and Abdel Ennaji

PSI Laboratory, Rouen FRANCE

Abstract. An incremental and Growing network model is introduced which is able to learn the topological relations in a given set of input vectors by means of a simple Hebb-like learning rule. First an overview of the most known models of Self-Organizing Maps (SOM) is given. Then we propose a new algorithm for a SOM which can learn new input data (plasticity) without degrading the previously trained network and forgetting the old input data (stability). We report the validation of this model on extensive experiments using a synthetic problem and the handwriting digit recognition problem over a portion of the NIST database.

1 Introduction

Data clustering is one of the most traditional and important issues in computer sciences. Cluster analysis aims at discovering groups and identifying meaningful distributions and patterns in data sets. Numerous clustering algorithms have been proposed in the literature, where four major categories of clustering algorithms are identified, namely hierarchical, partitional, density based, and grid based techniques [1, 2]. In recent years, due to emerging complex applications such as data and text mining, data clustering has attracted a new round of attention [3]. One of the main challenges in the design of efficient and robust clustering algorithms is that, in many applications, new data sets are continuously added into an already huge database. One way to tackle this challenge is to introduce a clustering algorithm that operates incrementally. However, few works in the development of incremental clustering algorithms are related in the literature [4, 5]. This paper presents the Incremental Growing Neural Gas algorithm (IGNG), a new incremental clustering algorithm for numerical data sets based on self-organizing maps principles. As the experiments conducted in this study reveal, the IGNG algorithm performs as well as the most known clustering algorithms in the static clustering way, and give superior performances for an incremental learning in comparison with the Growing Neural Gas algorithm [6]. In the next section, an overview of the most important self organizing maps is given. Section 3 discusses how the IGNG algorithm works. Section 4 reports some experimental results and comparisons conducted on a synthetic data set and the problem of handwriting digit recognition. Tests are conducted in order to analyse the performance of this algorithm in different situations including the problem of data visualisation and more important data sets in high dimension. Comparisons are given in the supervised meaning by labeling the obtained self organized map using the learning set labels. Finally, concluding remarks and future work are given in Section 5.

2 An overview of Self-organizing Map

We first would like to state the properties shared by all models described below. The network structure is a graph consisting of a set of nodes (units) A and a set of edges N connecting the nodes. Each unit c has an associated position (or reference vector) w_c in the input space. Adaptation, during the learning, of the reference vectors is done by moving the position of the nearest (or the "winning") unit (neuron) c_1 and its topological neighbors in the graph toward the input signal. For an input signal ξ the nearest unit c_1 is :

$$c_1 = \min_{c \in A} dist(\xi, w_c) \tag{1}$$

and the position update is made following this:

$$\Delta w_c = \epsilon(t) h_{c,c_1} \|\xi - w_c\| \tag{2}$$

where $\epsilon(t)$ is the adaptation step and h_{c,c_1} is a neighborhood function.
We can differentiate two kinds of SOM, Static Self-organizing Map and Growing Self-organizing Map. The Static SOM have a pre-defined structure which is chosen *a priori* and does not change during the parameter adaptation. In the opposite, Growing SOM have no pre-defined structure which is generated by successive addition (and possibly deletion) of nodes and/or connections.

Static Self-organizing Map : The **Kohonen's Self-Organizing Feature Map** (SOFM) [7] method considers an hyper-rectangular structure on the graph. Adaptation steps as described above are performed with:

$$\epsilon(t) h_{c,c_i} = \begin{cases} f(t) & \text{if } c \in N_{c_1}(t) \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where $0 \leq f(t) \geq 1$ is a decreasing monotonous function and:

$$N_{c_1}(t) = \{c \in A / \text{there is a path between } c_1 \text{ and } c \text{ that is smaller than } R_{c_1}(t)\}$$

$R_{c_1}(t)$ is a decreasing integer function. the decreasing behavior of both function f and neighborhood area of the winning unit during learning, makes it possible to coarsely explore the input space at the beginning of the training process, and to carry out a refinement of the unit position in the final phase.

Self-organizing Map generated by a constructive process : The **Growing Cell Structures** (GCS) model [8] has a structure consisting of *Hypertetrahedrons* with a dimensionality chosen in advance. A k-dimensional hypertetrahedron is a k-polyhedron having only $k + 1$ vertices. For examples, k is respectively to 1, 2 and 3 for lines, triangles and tetrahedrons. The model is initialized with one hypertetrahedron. Adaptation steps as described above are performed with:

$$\epsilon(t) h_{c,c_i} = \begin{cases} \epsilon_b & \text{if } c = c_i \\ \epsilon_n & \text{if there is an edge between } c \text{ and } c_i \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

At each adaptation step local error information is accumulated at the winning unit c_1.

$$\Delta E_{c_1} = \|w_{c_1} - \xi\|^2$$

After a given number λ of these adaptation steps, unit q with the maximum accumulated error is determined and a new unit is inserted by splitting the longest edge emanating from q. Moreover, additional edges are inserted in order to have a structure with hypertetrahedrons. The GCS model has a fixed dimensionality, so it realizes a dimensionality-reducing mapping from the input space into a k-dimensional space. This can be used for data visualisation.

The **Growing Neural Gas** (GNG) model [6] does not impose any explicit constraints on the graph topology. The graph is generated and continuously updated by competitive Hebbian Learning [9]. The adaptation of reference vectors is identical in GNG and in GCS. After a fixed number λ of adaptation steps, unit q with the maximum error is determined and a new unit is inserted between q and its neighbor p in the graph which has the maximum error. Error variables of q and p are re-distributed with the new unit. The topology of a GNG network reflects the topology of the input data distribution and can have different dimensionalities in different parts of the input space. For this reason a visualization is only possible for low-dimensional input data.

3 Incremental Growing Neural Gas

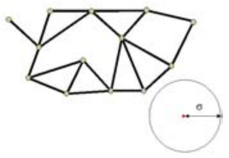
In the same way of the Growing Neural Gas model, our approach does not impose any explicit constraint on the graph which is generated and continuously updated by Competitive Hebbian Learning [9]. On the contrary to the network previously described, two kind of neurons are distinguished in our approach: mature neurons, and *embryos* neurons.

In addition, connections or edges between neurons are created dynamically during learning following the principles of the CHL algorithm. And both neurons and edges are associated to an age which is set to zero when created and updated during learning. When a new neuron is inserted, it is an *embryo* neuron and its age is set to zero.

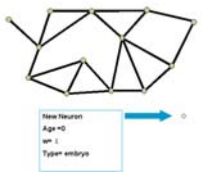
Initially, the graph is empty. At each iteration, we search the winning unit following (eq. 1). Then, like in the Adaptive Resonance Theory (ART), we perform the vigilance test of eq. 5 in order to decide if the winning unit is close enough to the input signal.

$$dist(\xi, w_c) \leq \sigma \tag{5}$$

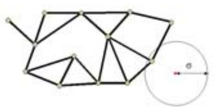
If the graph is empty, or the winning unit does not satisfy the vigilance test, we add a new *embryo* neuron with $w_{new} = \xi$ and the iteration is finished. This case is shown in figure 3 a) and b). Figure 3(a) illustrates an IGNG network with new data which have to be learned (in dimension 2). This data is too far from the winning unit, so a new *embryo* neuron (fig 3(b)) is created. The reference vector



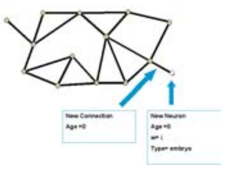
(a) A new training sample



(b) creation of a new neuron



(c) A new training sample



(d) creation of a new neuron

of this new unit is the position of the data in the input space. If the vigilance test is satisfied for the winning unit, we perform it to the second-nearest unit. If there is only one unit in the graph, or the test is not satisfied for the second nearest unit, a new *embryo* neuron is added with $w_{new} = \xi$. Figure 3 c) and d) shows this case. In this figure the new data satisfy the vigilance test with its winning unit, but it is too far from the second-nearest neuron, so we create a new neuron *embryo* which is connected to the winning unit (fig 3(d)). The reference vector of the new unit is the position of the data in the input space.

When the two nearest units satisfy the vigilance test, the reference vector of the units are adapted as in equations 2 and 4.

The learning algorithm is then continued throughout adaptation of both neurons and edges. If the connection between the two nearest units does not exist, an edge is created. Otherwise, the age of this edge is set to zero. On the other hand, the age of all other edges connected to the winning unit is incremented. Afterward, we remove edges with an age larger than a_{max}, and if this leads to an isolated mature neurons (without connections), we remove them as well.

Then, we increment the age of all direct neighbor neurons of the winning unit. Consequently, if a giving *embryo* neuron has an age larger than a_{mature}, this unit becomes a mature neuron. The final graph is only constituted of mature neurons, *embryos* neurons are useful only for the training.
This process is detailed in algorithm 1.

4 Experimental results

In this section, we will report some experimental results to demonstrate the general behavior and performances of our model. For data visualization, a synthetic problem with low-dimensional input data, which allows to show the limits of the GNG model in incremental learning, is used.

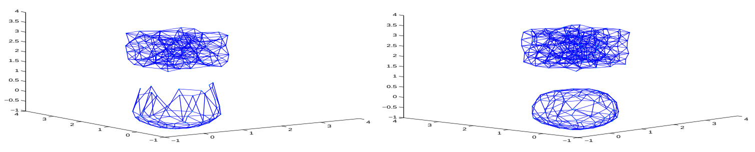
Handwritten Digit Recognition : In order to validate our model for high-dimensional problems, we report some results obtained for the handwriting digit recognition problem over the NIST database 3. The feature vector considered is constituted by the 85 (1+4+16+64) gray levels of a 4-level-resolution pyramid [10]. In this experiment, neurons are labeled for a classification task. A simply majority vote rule (in the supervised meaning) of the learning data that have activated each neuron during the training process is used for this. Two subsets of the NIST database of respectively 2626 for training and 2619 other digits for testing are used in this experiment. This experiment is made to compare our approach to the GNG model and LVQ approach for a classification problem. The three network are compared using a NIST database portion described below. A learning cycle corresponds to a presentation of all the training database samples to the network. For incremental learning, the used portion of the NIST database has been split in four parts. Each model has been trained then on one part after another . Table 1 shows that our approach performs a faster learning (5

	Cycles	λ	ϵ_b	ϵ_n	σ	a_m	Units	offline	incr
LVQ	50	-	-	-	-	-	330	89.65%	-
GNG	50	400	0.1	0.006	-	-	330+0	91.44%	81.29
IGNG	**10**	-	0.01	0.002	2.7	450	313+9	**91.71%**	**90.18%**

Table 1: Recognition rate of the three networks in the portion of the NIST database

times less) than the GNG network for a comparable accuracy, and with a little less number of neurons. Note that in this experiment, 9 embryo neurons are generated during the learning.

A synthetics Problem : The synthetic problem we consider here is composed of two classes. The first class has a spherical distribution, while the second one has a cubic shape. The network was first trained with the spherical class. Then, to produce incremental learning, a data subset with only samples from the second class was presented to the trained network.



(e) GNG (f) IGNG

Figure 1: GNG (a) and IGNG (b) behavior in incremental learning

Fig.1 shows the result obtained with the GNG model. On this figure, we can observe the loss of the learned topology with the GNG model reflecting an important distortion of the previously learned class. Such a phenomena is avoided with our approach which allows a good behavior concerning the dilemma stability/plasticity in an incremental learning context.

5 Conclusion

The "Incremental Growing Neural Gas" network presented in this paper is able to make explicit the important topological relations in a given distribution $P(\xi)$ of input signals. An advantage over other classical models is the stability of our network in incremental learning. Experiments carried out for low and high-dimensional problems demonstrate this stability. These results support the conclusion that this contribution can be considered as a response of the well known dilemma plasticity/stability in machine learning field.

Possible applications of our model are data visualization (only in low-dimensional input space), clustering and supervised learning (in the same way that LVQ).

Another promising way and future developments of this approach concern its combination with several supervised classifiers allowing the design of an incremental supervised decision system. Good and promising results have been already obtained in [11] with a multi-classifiers system which is managed by an IGNG network.

References

[1] N. Jain, A. K.and Murty and P. J. Flyn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[2] M.F. Janowitz. A combinatorial introduction to cluster analysis. Technical report, Classification Society of North America, 2002.

[3] P. O. Duda, P. E. Hart, and D.G. Storck. Pattern classification. New York, 2001.

[4] B. Fritzke. Growing self-organizing networks - why ? In *ESANN*, pages 61–72, 1996.

[5] A. Ribert, A. Ennaji, and Y. Lecourtier. An incremental hierarchichal clustering. In *Proc. of Vision Interface Conf*, pages 586–591, 1999.

[6] B. Fritzke. A growing neural gas network learns topologies. In *Advances in Neural Information Processing Systems 7*, pages 625–632. 1995.

[7] T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69, 1982.

[8] B. Fritzke. Growing cell structures — A self-organizing network for unsupervised and supervised learning. *Neural Networks*, 7(9):1441–1460, 1994.

[9] T. Martinetz. Competitive Hebbian learning rule forms perfectly topology preserving maps. In *Proc. ICANN'93*, pages 427–434. Springer, 1993.

[10] D.H. Ballard and C. M. Brown. *Computer Vision.* Prentice Hall, 1982.

[11] Y. Prudent and A. Ennaji. Clustering incrémental pour un apprentissage distribué. In *Conférence CAp 2004*, 2004.