DB-GNG: A constructive Self-Organizing Map based on density

Alexander Ocsa, Carlos Bedregal, Ernesto Cuadros-Vargas

***Abstract*— Nowadays applications require efficient and fast techniques due to the growing volume of data and its increasing complexity. Recent studies promote the use of Access Methods (AMs) with Self-Organizing Maps (SOMs) for a faster similarity information retrieval. This paper proposes a new constructive SOM based on density, which is also useful for clustering. Our algorithm creates new units based on density of data, producing a better representation of the data space with a less computational cost for a comparable accuracy. It also uses AMs to reduce considerably the Number of Distance Calculations during the training process, outperforming existing constructive SOMs by as much as 89%.**

I. Introduction

The volume of data that needs to be managed is growing every day and increasing in complexity, involving DNA sequences, video and multimedia information. Applications require even more efficient and faster techniques, over several computer fields, such as information retrieval, video compression, bioinformatics and data mining.

Database and Information Retrieval communities use Access Methods due to their ability to build data structures that easily manage and organize large datasets in an efficient way even in high dimensions.

On the other hand, SOMs [1] have been widely used to solve clustering problems due to their unsupervised classification capacity. Within the SOM networks, Growing Neural Gas (GNG) [2] stands out because of its incremental clustering algorithm. Recent investigations of GNG networks such as Incremental Growing Neural Gas (IGNG) [3] and TreeGNG [4] improve the performance of the incremental algorithm but the dependency on initial parameters is still critical. Some problems with the SOM networks are the difficulty in supporting specific queries such as k-nearest neighbor or range queries for large datasets. Another drawback is the expensive training process due to the sequential approach; although parallel architectures can be used, a sequential search is still applied.

Working with high dimensional data, the learning process could be expensive because a large number of distances need to be calculated for each data point during the training process. Besides the computational complexity, factors such as database size, dimensionality of data and even the distance function could lead to a high computational cost [5], [6].

Evaluation of similarity in SOMs could represent a high cost compared with CPU time and I/O time, consequently, the challenge is to minimize the Number of Distance Calculations (NDC) [7]. For this reason our experiments will be based on the NDC as a measure of efficiency.

Recent studies such as Hybrid SAM-SOM and MAM-SOM [5] incorporate Access Methods into a SOM network to accelerate the building process. SAM-SOM* and MAM-SOM* [6] techniques, in addition to a faster building process, support similarity queries and show interesting properties. However, these techniques for clustering generation lack specificity.

This paper proposes a new incremental clustering algorithm, taking advantage of the strengths of the Hybrid SAMSOM, MAMSOM* and IGNG, technique that besides working faster than previous approaches, this algorithm enables faster similarity queries.

The paper is organized as follows: Section II presents a review of previous work. Section III describes the proposed technique. Section IV shows the experimental results. Finally, Section V presents the conclusions of the paper.

II. Previous Work

A drawback of Kohonen's SOM is the static topology of the network, besides the high training cost. Incremental and hierarchical models try to solve this problem by creating dynamic topologies (as GNG [2]) or by reducing the training time needed through the use of hierarchical approaches (as TreeGNG [4]). Growing Neural Gas is a worthy representative of incremental networks, but the training time and the resulting network both depend on the parameter λ (the number of patterns that must be presented in order to insert a new unit into the network). The total NDC needed to build a network of n units is determined by Equation 1. If λ is set with a high value the construction of the network tends to be slow; on the other hand, a small value of λ would lead to the network deformation [6].

$$NDC_{GNG} = \lambda \frac{n^2 - n - 2}{2} \quad (1)$$

Alexander Ocsa is with the San Agustin National University, Av. Independencia Arequipa Peru.(email: aocsa@ieee.org).

Carlos Bedregal is with the San Pablo Catholic University, Av. Salaverry 301 Arequipa Peru.(email: cbedregal@ieee.org).

Ernesto Cuadros-Vargas is with the San Pablo Catholic University, Av. Salaverry 301 Arequipa Peru, and the Peruvian Computer Society. (email: ecuadros@spc.org.pe).

Improvements of the GNG algorithm such as IGNG [3] eliminate the dependency on λ, accelerating the building process of the network. IGNG establishes a parameter σ which defines a search radius to create a new unit. As it happens to λ in GNG, if σ is set with a high value, IGNG will create a small number of clusters and the topology of the data will not be represented correctly; if σ is set too low, many units could be created and the algorithm would not have enough time to adapt to the input space. Algorithm 1 shows the general construction of an Incremental Growing Neural Gas (IGNG) network [3].

Algorithm 1: IGNG Algorithm

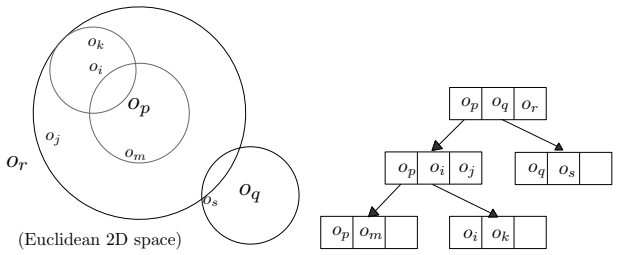
1: For each input pattern ξ
- Find the closest unit c_1;
- If the distance $d(\xi, c_1) > \sigma$ then create a new unit with $w = \xi$;
- Else, find the second closest unit c_2;
  - If the distance $d(\xi, c_2) > \sigma$ then create a new unit with $w = \xi$ and connect it with c_1;
  - Else, update weights for c_1 and its neighbors, and increase the age of its connections;

New techniques incorporate Access Method (AM) into SOMs, accelerating the training process of the network due to the reduction of the number of distance calculations and giving the network capacity to perform to similarity queries [7].

A. Metric and Spacial Access Methods

Metric Access Methods (MAM) index data objects in a metric space based on similarity criteria. Metric spaces are defined as a set of objects and a metric distance function that measures the dissimilarity among these objects and satisfies properties such as positiveness, symmetry, reflexivity and triangular inequality. Some classic MAMs are described in [8]. For our experiments the following MAMs are considered:

- **Slim-Tree** [9]. Slim-Tree is an evolution of the M-Tree [10], which is the first dynamic MAM that allows the construction of a tree without knowing the entire dataset a priori. Slim-Tree divides the space in regions that store subtrees which in turn contain more subtrees. Subtrees may be overlapped, partitioning the metric space in not necessarily disjoint regions. This MAM grows bottom-up storing the objects into leaf nodes as in B^+-Tree [11] and R-Tree [12]. With an algorithm to split nodes based on the Minimal Spanning Tree (MST), Slim-Tree has less computational cost without sacrificing search performance. It also introduces an overlapping measure and the algorithm Slim-Down [9] to reduce the overlap between nodes.
- **DBM-Tree** [13]. One of the latest MAM, the DBM-Tree, operates in a similar way to a Slim-Tree. It was the first Access Method which proposed to relax the height of the tree in regions with high density of data in order to minimize the overlap of nodes. This approach reduces the number of distance computations without affecting the number of disk access. Figures 1(a) and 1(b) show an example of this structure for two dimensional data.



(a) Metric regions represented by DBM-Tree

(b) DBM-Tree internal representation

Fig. 1. DBM-Tree Representations

In the case of Spatial Access Methods (SAM) the input data is described by feature vectors such as $(x_1, x_2, ..., x_n)$. Some classic SAM are explained in [14], [15]. In this article we use R-Tree to speed up the network training process. This SAM is briefly explained below.

- **R-Tree** [12]. This is the first non point-based SAM, able to index not only vectors but also geometrical objects. R-Tree could be considered a generalization of B-Trees [11] to index multidimensional data. In this SAM the information is stored in the leaves and each upper level has the information about the Minimum Bounding Rectangle (MBR) necessary to contain all its children nodes. Before the tree is built, it is necessary to define the maximum number of objects that each node can contain. Figures 2 and 3 show a R-tree with 3 maximum objects per node.

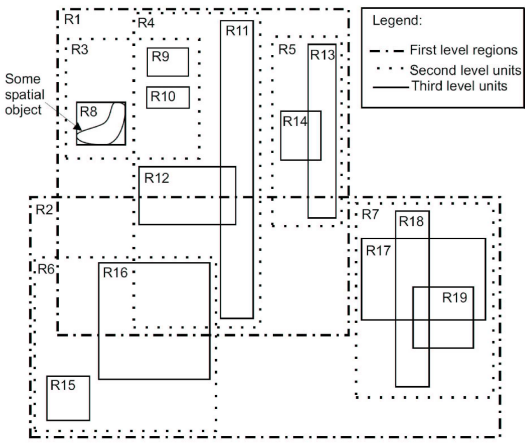


Fig. 2. MBRs represented by a R-Tree [5]

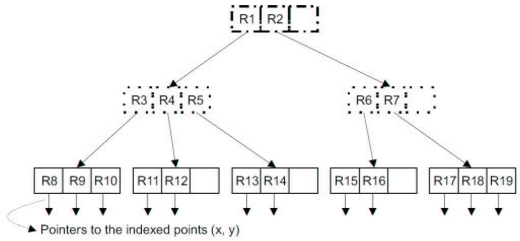


Fig. 3. R-Tree internal representation [5]

III. Density Based Growing Neural Gas

The proposed algorithm of Density Based Growing Neural Gas (DB-GNG) uses a radius parameter for training the network and for the insertion of units as in IGNG [3].

Additionally DB-GNG considers the existence of a dense data region as new criterion to create units. Our algorithm uses an Access Method (AM) to determine if a new pattern ξ presented to the network is located in a region with more than ϕ patterns within a σ radius from ξ.

When this radius parameter is smaller, units are created where data is more concentrated, identifying only the more dense regions of the space, in contrast with IGNG that would insert units indiscriminately.

Algorithm 2 shows the process of construction of the DB-GNG network: here AM_1 is the Access Method used to index the neurons weight vector and to find the two Best Matching Units (BMUs), and AM_2 is the Access Method where all the input patterns are organized during training. On each epoch input patterns are inserted again into AM_2 in order to improve the density criterion. This additional insertions reduce the training process and improve the classification capacity of the network as it will be observed in the experiments.

Density Based Growing Neural Gas (DB-GNG) does not create units in regions less populated because before inserting, the algorithm verifies if the new unit is located in a dense region of the space, thus obtaining a better representation of the data distribution.

The use of AMs to find the winning neuron dramatically accelerates the training process of the network, as demonstrated in [5]. Furthermore, in order to give support to specific queries such as nearest neighbors, DB-GNG has a second Access Method AM_2 to organize patterns as they are presented. AM_2 is also used to detect dense regions.

Similarity queries in DB-GNG are more efficient than using single AMs. The most common technique that Access Methods use to solve nearest neighbor queries combines the algorithm of range query (which retrieves the elements within a radius) and the branch and bound technique [16].

[1]the stop criterion could be the number of iterations, number of epochs, maximum number of units, etc.

Algorithm 2: Density Based Growing Neural Gas training algorithm

while a stop criterion is not met[1] **do**
  Insert the next input pattern ξ into the AM_2 ;
  Find the two closest units c_1 and c_2 using AM_1;
  if distance $d(\xi, c_2) \leq \sigma$ **then**
    Increment the age of all edges of c_1;
    $w_{c_1} += e_b(\xi - w_{c_1})$; {weight update}
    $w_n += e_n(\xi - w_n)$; $\{\forall n \in$ neighbors of $c_1\}$
    Update the subtree in AM_1 affected with the weight update;
    Remove edges with an age bigger than a_{max} and remove neurons with no edges;
  else
    $S = RangeQuery(\xi, \sigma)$ using AM_2;
    if $|S| \geq \phi$ {detect if the region is dense} **then**
      Create a new t unit with $w_t = \xi$;
      Connect t with c_1 and c_2;
      if $d(c_1, c_2) \geq max(d(c_1, t), d(c_2, t))$ **then**
        Remove the connection between c_1 and c_2;
      end if
    end if
  end if
end while

To retrieve the k nearest neighbors, AMs are initialized with an infinite radius and this radius is reduced gradually by using the branch and bound technique until the k required elements are covered.

In order to reduce the cost of the query, DB-GNG establishes the initial radius in kNN queries through the codevectors generated during the training process, as it can be observed in Algorithm 3. This reduces the cost and time of queries.

Algorithm 3: DB-GNG kNN algorithm

Find the two closest units to ξ, c_1 and c_2 using AM_1;
Define a radius $r = d(\xi, c_2) \times \upsilon^k$; {$\upsilon$ modifies the radius for queries where $k > \phi$.}
Solve the query using AM_2 with initial radius r;

IV. Experiments

Based on this idea, four groups of experiments are presented using the AMs Slim-Tree [9], DBM-Tree [13] and R-Tree [12]. In the first group of experiments the building process of our approach is compared to another constructive network, each technique measured in terms of Number of Distance Calculations (NDC). The second group of experiments compares the proposed network to Access Methods in terms of the NDC performed in similarity queries. The third group compares our approach

to another two constructive networks for a classification problem. The last group of experiments compares visually the resulting networks of three incremental algorithms for a clustering problem. All experiments were implemented using Microsoft Visual C++ 6.0 on a PC 2.0 GHz 512 RAM running Microsoft Windows XP.

Five datasets were used for our experiments:

- COVERT: This dataset contains a sample set of 50000 vectors from the original dataset of 581012 vectors in 54-d of as Forest Covertype data. The file was obtained from *UCI-Irvine repository of machine learning databases and domain theories*[3];
- DIGITS: This dataset was created for the pen-based recognition of handwritten digits of 44 writers; it contains 10992 16-d vectors. The file was obtained from *UCI-Irvine repository of machine learning databases and domain theories*[4];
- NURSERY: This dataset contains 12960 8-d vectors derived from a hierarchical decision model developed in order to rank applications for nursery schools. The file was obtained from *UCI-Irvine repository of machine learning databases and domain theories*[5];
- OPTIDIGITS: This dataset contains 5620 64-d vectors obtained from the optical recognition of handwritten digits from a total of 43 writers. The file was obtained from *UCI-Irvine repository of machine learning databases and domain theories*[6];
- 2D SYNTHETIC: This dataset contains 1000 2-d synthetic points between $(0,0)...(5,5)$.

The Access Method used to find the winning neurons AM_1 in DB-GNG techniques was R-Tree [12]. For all the experiments σ was set with a value close to the standard deviation of the dataset, and υ was set to $\frac{\sigma}{8}$.

The techniques compared in the first group of experiments are: IGNG [3], DB-GNG using Slim-Tree [9] and DB-GNG using DBM-Tree [13], applied in the COVERT, DIGITS and NURSERY datasets.

Figures 4, 5 and 6 illustrate the first group of experiments in terms of the accumulated Number of Distance Calculations (NDC).

Table I shows the results for the COVERT dataset. For both IGNG and DB-GNG, the parameter σ was set to 497.536, and for DB-GNG ϕ was set to 5.

Table II shows the results for the DIGITS dataset. For both IGNG and DB-GNG, the parameter σ was set to 35.24, and for DB-GNG ϕ was set to 7.

Table III shows the results for the NURSERY dataset. For both IGNG and DB-GNG, the parameter σ was set to 1.054, and for DB-GNG ϕ was set to 7.

[3] `ftp://ftp.ics.uci.edu/pub/machine-learning-databases/covtype/`

[4] `ftp://ftp.ics.uci.edu/pub/machine-learning-databases/pendigits/`

[5] `ftp://ftp.ics.uci.edu/pub/machine-learning-databases/nursery/`

[6] `ftp://ftp.ics.uci.edu/pub/machine-learning-databases/optdigits/`

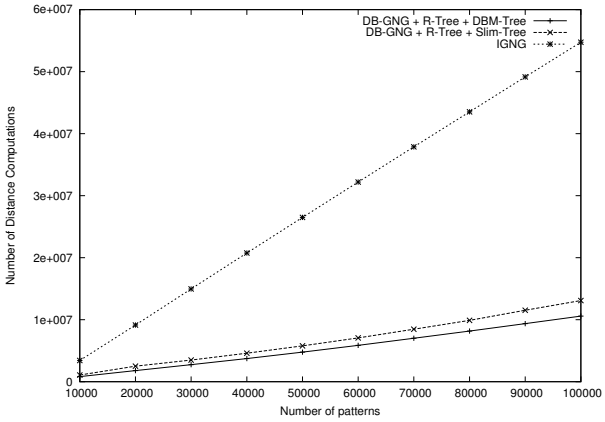


Fig. 4. Accumulated NDC per pattern presented using the COVERT $(54-d)$ dataset.

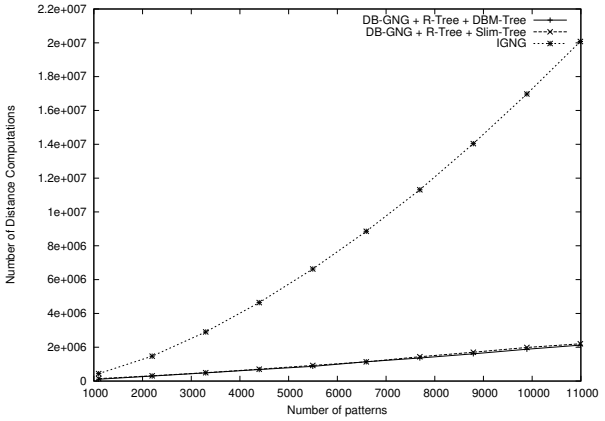


Fig. 5. Accumulated NDC per pattern presented using the DIGITS $(16-d)$ dataset.

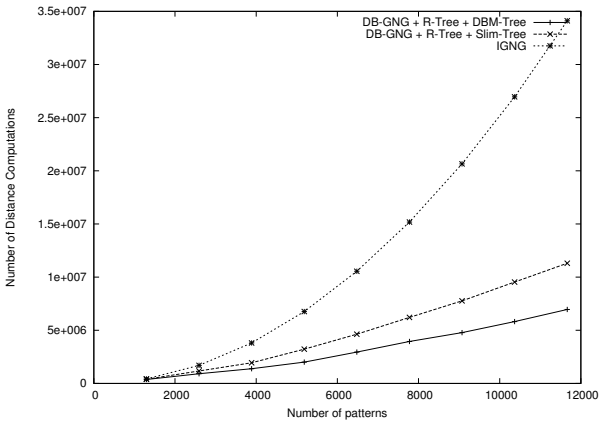


Fig. 6. Accumulated NDC per pattern presented using the NURSERY $(8-d)$ dataset.

TABLE I
RESULTS FOR THE COVERT $(54-d)$ DATASET

Technique	# of units	NDC	$\%gained$
IGNG	558	54745299	
DB-GNG Slim-Tree	6	13087814	76.1%
DB-GNG DBM-Tree	6	10579868	80.7%

TABLE II
RESULTS FOR THE PENDIGITS $(16-d)$ DATASET

Technique	# of units	NDC	$\%gained$
IGNG	2890	20084976	
DB-GNG Slim-Tree	96	2206760	89.0%
DB-GNG DBM-Tree	96	2126153	89.4%

TABLE III
RESULTS FOR THE NURSERY $(8-d)$ DATASET

Technique	# of units	NDC	$\%gained$
IGNG	6490	42095871	
DB-GNG Slim-Tree	1405	12710986	69.8%
DB-GNG DBM-Tree	1405	7958717	81.1%

As it can be observed in the first group of experiments, DB-GNG reduces by up to 89% the NDC in comparison to IGNG. This difference is due to because the use of AMs and the small number of units that DB-GNG creates as a result of its density criterion, which considers the existence of a dense data region to create new units.

It must be noted that DB-GNG also accumulates the distance calculations needed by the Access Methods AM_1 (R-Tree) and AM_2 (Slim-Tree or DBM-Tree) during their construction, as well as the distance calculations performed while verifying the regions's density.

Results for the second group of experiments are presented in Figures 7 and 8. These experiments compare the average Number of Distance Calculations (NDC) when performing kNN queries. The techniques compared are: DB-GNG using Slim-Tree, DB-GNG using DBM-Tree and Access Methods Slim-Tree [9] and DBM-Tree [13]. The parameters for each dataset (COVERT and DIGITS) were the same as in the first group of experiments.

In these experiments it can be seen that DB-GNG networks need less distance calculations to perform kNN queries than their counterparts using only Access Methods DBM-Tree and Slim-Tree.

For the fourth group of experiments incremental networks GNG, IGNG and DB-GNG using Slim-Tree are compared in a classification problem using the OPTIDIGITS $(64-d)$ dataset. A subset of 3823 samples were used for training and a subset of 1797 samples for testing. For GNG λ was set to 600, e_b to 0.01 and e_n to 0.002. For IGNG and DB-GNG σ was set to 30.04, e_b to 0.001 and e_n to 0.00002. Additionally for DB-GNG ϕ was set to 3.

Table IV shows that DB-GNG creates a faster network,

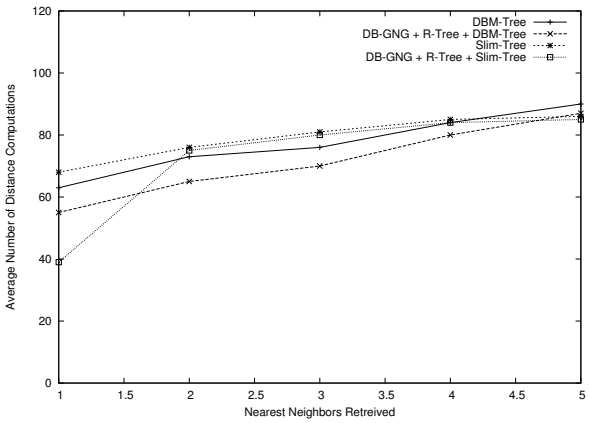


Fig. 7. Average NDC performing kNN queries using the COVERT $(54-d)$ dataset.

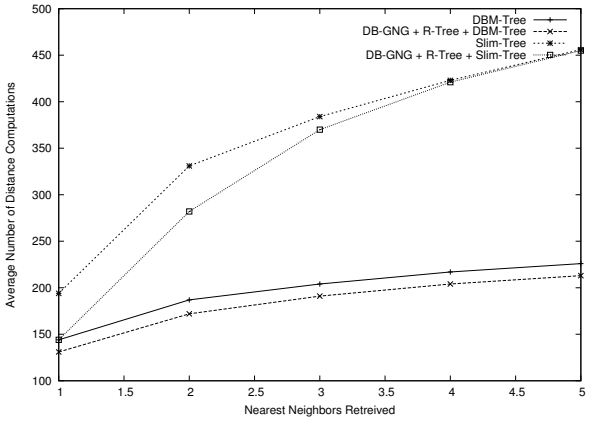


Fig. 8. Average NDC performing kNN queries using the DIGITS $(16-d)$ dataset.

TABLE IV
RECOGNITION RATE FOR THE OPTIDIGITS $(64-d)$ DATASET

Technique	Epochs	Units	Recognition	NDC
GNG	30	193	95.05%	12206650
IGNG	6	431	95.05%	11614569
DB-GNG	3	385	95.60%	5210035

with less number of epochs and less NDC than GNG and IGNG networks for a comparable accuracy.

Finally, Figures 9, 10 and 11 show GNG, IGNG and DB-GNG respectively using the 2D SYNTHETIC dataset. For the GNG network λ was set to 300, for IGNG and DB-GNG σ was set to 0.3, and for DB-GNG ϕ was set to 100. It is possible to compare visually our proposal DB-GNG with other incremental networks. Due to its density criterion to create units, DB-GNG produces a better representation of the input distribution in less iterations than IGNG and GNG.

V. CONCLUSIONS

The experiments showed that the Density Based Growing Neural Gas (DB-GNG) can be applied to cluster large databases. The performance of DB-GNG was up

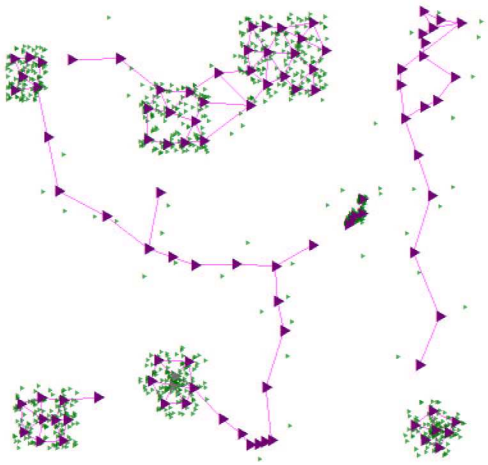


Fig. 9. Visualization of GNG $\lambda = 300$ using the 2D SYNTHETIC dataset. 102 units were created after 30 cycles.

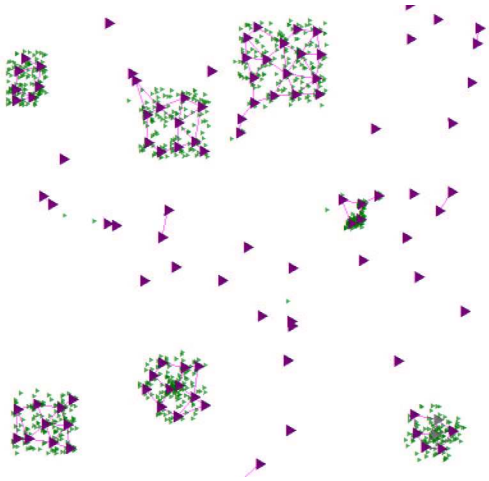


Fig. 10. Visualization of IGNG $\sigma = 0.3$ using the 2D SYNTHETIC dataset. 131 units were created after 15 cycles.

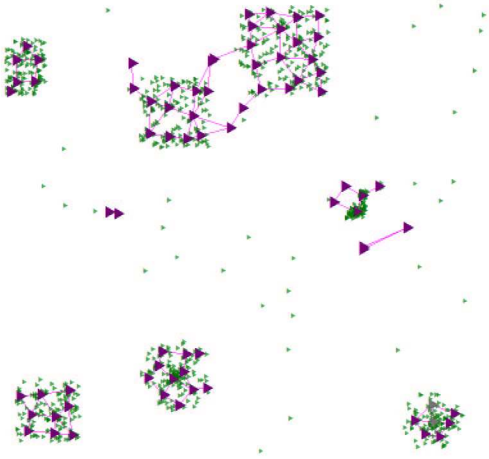


Fig. 11. Visualization of DB-GNG $\sigma = 0.3$, $\phi = 100$ using the 2D SYNTHETIC dataset. 77 units were created after 7 cycles.

to 89% better than the sequential IGNG as a result of the reduced number of units in its final network. This difference is also due to Access Methods which considerably reduce the Number of Distance Calculations needed during the training process.

Introducing a density criterion to create units, DB-GNG does not locate units in regions with low density, obtaining a better representation of the data space.

For classification problems, our approach proved to be as accurate as other incremental networks, while performing a faster training process.

For information retrieval, DB-GNG was demonstrated to be more efficient than single Access Methods, due to the use of codevectors to estimate selectivity for nearest neighbors queries.

References

[1] T. Kohonen, "Self-organized formation of topologically correct feature maps," pp. 509–521, 1988.

[2] B. Fritzke, "A growing neural gas network learns topologies," in *Advances in Neural Information Processing Systems 7*, G. Tesauro, D. S. Touretzky, and T. K. Leen, Eds. Cambridge MA: MIT Press, 1995, pp. 625–632.

[3] Y. Prudent and A. Ennaji, "A k nearest classifier design," *ELCVIA*, vol. 5, no. 2, pp. 58–71, 2005.

[4] K. Doherty, R. Adams, and N. Davey, "TreeGNG - hierarchical topological clustering." in *ESANN*, 2005, pp. 19–24.

[5] E. Cuadros-Vargas and R. F. Romero, "The SAM-SOM Family: Incorporating Spatial Access Methods into Constructive Self-Organizing Maps," in *Proc. International Joint Conference on Neural Networks IJCNN02*. Hawaii, HI: IEEE Press, 2002, pp. 1172–1177.

[6] E. Cuadros-Vargas and R. A. F. Romero, "Introduction to the SAM-SOM* and MAM-SOM* families," in *International Joint Conference on Neural Networks (IJCNN05)*. IEEE, July 2005.

[7] C. Bedregal and E. Cuadros-Vargas, "Using large databases and self-organizing maps without tears," in *International Joint Conference on Neural Networks (IJCNN06)*. IEEE, July 2006.

[8] E. Chávez, G. Navarro, R. Baeza-Yates, and J. Marroquín, "Proximity searching in metric spaces," *ACM Computing Surveys*, vol. 33, no. 3, pp. 273–321, Sept. 2001.

[9] J. Caetano Traina, A. J. M. Traina, B. Seeger, and C. Faloutsos, "Slim-trees: High performance metric trees minimizing overlap between nodes," in *Proc. of the 7th International Conference on Extending Database Technology EDBT00*. London, UK: Springer-Verlag, 2000, pp. 51–65.

[10] P. Ciaccia, M. Patella, and P. Zezula, "M-tree: An efficient access method for similarity search in metric spaces," in *The VLDB Journal*, 1997, pp. 426–435.

[11] D. Comer, "Ubiquitous b-tree," *ACM Comput. Surv.*, vol. 11, no. 2, pp. 121–137, 1979.

[12] A. Guttman, "R-Trees: A dynamic index structure for spatial searching," in *Proc. of Annual Meeting, Boston, Massachusetts SIGMOD84, June 18-21, 1984*, B. Yormark, Ed. ACM Press, 1984, pp. 47–57.

[13] F. J. T. C. Marcos R. Viera, Caetano Traina Fr. and A. J. Traina, "DBM-tree: A dynamic metric access method sensitive to local density data," Brazilian Symposium on Databases, 2004.

[14] H. Samet, *Spatial Data Structures*, Kim W ed., ser. Modern Database Systems. Reading, Addison-Wesley/ACM, 1995.

[15] V. Gaede and O. Günther, "Multidimensional Access Methods," *ACM Computing Surveys*, vol. 30, no. 2, pp. 170–231, 1998.

[16] L. Micó;, J. Oncina, and R. C. Carrasco, "A fast branch & bound nearest neighbour classifier in metric spaces," *Pattern Recogn. Lett.*, vol. 17, no. 7, pp. 731–739, 1996.