Local Variance Driven Self-Organization for Unsupervised Clustering

Matthew Kyan[1], Ling Guan[2]
[1]*University of Sydney, Australia,* [2]*Ryerson University, Canada*
[1]*mkyan@ee.usyd.edu.au,* [2]*lguan@ee.ryerson.ca*

Abstract

We propose a new, novel unsupervised clustering technique based on traditional Kohonen self organization, Competitive Hebbian Learning (CHL), and the Hebbian based Maximum Eigenfilter (HME). This method fits into the family of dynamic self-generating, Self-Organizing Map (SOM) algorithms. The approach uses a vigilance based, global parsing strategy as a guide for the hierarchical partitioning of an underlying data distribution into a set of dominant prototypes: each consisting of a dual memory element for the online estimation of both position and maximal local variance. A co-operative scheme exploits the interplay between global vigilance and maximal local variance such that an informed choice may be made regarding insertion sites for new nodes into the Map. The network is related to Self-Organizing Tree Maps (SOTM), Growing Neural Gas (GNG) and their variants. A framework is presented and performance demonstrated against GNG.

1. Introduction

The goal of Unsupervised Learning, as applied to the field of pattern recognition, is to essentially formulate or discover significant patterns or features in a given set of data, *without* the guidance of a teacher. The patterns are usually stored as a set of prototypes or clusters: representations or groupings of *similar* data. In describing an unknown set of data, such techniques find much application across a wide range of industries, particularly in bioinformatics (clustering of genetic data, protein structure), image processing (segmentation, image retrieval), and other applications that warrant a significant need for data mining.

Popular approaches to unsupervised clustering generally fall into groups based on either: *squared-error* (e.g. K-means and Fuzzy K-means), *mixture decomposition* (e.g. Gaussian Mixture Models), *hierarchical agglomeration* (e.g. Single- or Complete-Link, SNN), or *neural networks* based on Self-Organization [1].

Unlike the other techniques, Self-Organization has a basis in the biological process of *associative* memory, and is inherently related to unsupervised learning. As such, it has been instrumental in the development of many foundational techniques such as Kohonen's *Self-Organizing Map* (SOM) which continue to influence modern data mining research.

Associative connections linking prototypes within SOM based clustering algorithms are generally responsible for their innate ability to infer an *ordered* or *topologically preserved* mapping of the underlying data space. Associations between nodes are advantageous as they help guide the evolution of such networks, and may also assist in formulating post-processing strategies or for extracting higher-level cluster properties (e.g. inter-cluster relationships).

In typical SOM based approaches, topology is imposed rather than inferred (via a fixed predefined lattice of nodes). Unsupervised learning however, is ill-posed: the nature of the underlying data is unknown, thus it is difficult to infer what an appropriate number of classes might be, or how they should be related: a common hurdle for all clustering techniques.

Dynamically generating SOM networks attempt to address this issue by formulating a set of hierarchical associations as they grow to represent the data space.

In this paper, we explore and develop a novel means of dynamically generating a network of self assembled nodes toward a network of cluster prototypes. We propose a novel approach related to the *Self-Organizing Tree Map* (SOTM) [2], namely, the *Self-Organized Hierarchical Variance Map* (SOHVM). The methodology resembles Evolving Tree (ET) [3] & Growing Neural Gas (GNG) [4] related strategies, in that it partitions the input space in a divisive rather than agglomerative fashion, whilst building a topologically preserving set of associative connections between nodes. Unlike ET & GNG however, the approach draws on principals from

The 18th International Conference on Pattern Recognition (ICPR'06)
0-7695-2521-0/06 $20.00 © 2006 IEEE

Adaptive Resonance Theory (ART), wherein vigilance considerations made whilst parsing the input space govern the insertion of new nodes into a dynamically growing network structure. A hierarchical control function oversees this process, allowing for a globally sensitive, top-down partitioning of the feature space. During this process, the algorithm works to identify appropriate clusters and cluster associations, at progressive levels of granularity.

2. Foundations

2.1. Competitive Hebbian Learning

The principal of *Competitive Hebbian Learning* (CHL) [5], was first introduced as a means of strengthening and weakening associations such that topological connections form dynamically between nodes that tend to span the more dense regions of the underlying data as they are discovered. This is achieved by forming a connection between two nodes that are found to be most representative of any given input: akin to triggering or refreshing a correlation between the two nodes. An edge ageing scheme is used to weaken connections formed at an early state, but not re-affirmed later due to network changes. In the proposed model, CHL allows the network to evolve in terms of its topology: influencing how information is imparted to neighbouring nodes – the network is thus more free from limitations on its plasticity that tend to come from any pre-assumed structure.

2.2. SOTM

In a standard SOTM, node insertion is driven by a top-down process that effectively explores the data space from the *outside-in*. An ellipsoid of *significant similarity* [2], forms a global vigilance threshold $H(t)$ that is used to evaluate the proximity of each new input sample, against the closest existing prototype (winning node) currently in the network. A hard decision is made: if the input is distant from the winner beyond the threshold, a new prototype is spawned, and the process continues with the next input. Otherwise, the input is deemed *significantly similar* and the winner is updated toward the input.

This blind approach associates spawned nodes with their winners, progressively forming a topological tree structure. By forcing $H(t)$ to decay from a large initial value, clusters discovered early will be far from one another: a condition that is relaxed as the data space becomes more partitioned. Due to the random nature in which samples are presented to the network during training, node generation can be erratic. This is not an issue during early stages, as nodes inevitably *track back* to regions of density. However, during later phases, this becomes problematic, as more nodes compete for the right to draw information from input samples. Thus a spawned node may become trapped and unable to sufficiently move back to a more dense location. Furthermore, the tree structure formed maintains a *loose* notion of topology: the *associations* between nodes exist at the time of node generation, however are not maintained in lower levels of the tree.

2.3. Hebbian Maximal Eigenfilter

When considering a new site for node generation, SOTM proximity measures alone do not give much credence to what is known so far, nor what might be the most *likely* regions of input space that *warrant* further exploration. What is desired is a more *informed* decision for node insertion.

In the newly proposed model, bottom-up information is extracted by a *hebbian-based maximum eigenfilter* (HME) which probes data in the vicinity of each prototype. Interplay between such probes and the top-down parsing via $H(t)$ allows for a semi-informed decision on where to insert new nodes.

The HME [6] offers a useful mechanism that fits well into the online framework of our proposed map. Oja showed that, for a simple perceptron (1), the use of a single Hebbian-type adaptation rule (2) for its synaptic weights (q_k) can evolve (3) into a filter for the first principal component of the input distribution:

$$y = \sum_{i=1}^{N_f} q_i x_i \tag{1}$$

$$q_i(n+1) = q_i(n) + \eta \cdot y(n) q_i(n) \tag{2}$$

$$q_i(n+1) = q_i(n) + \eta \cdot y(n)\left[\, x_i(n) - y(n) q_i(n) \,\right] \tag{3}$$

Where x is the input vector, y is the output of the perceptron, and η is a learning rate. As the perceptron is trained with samples from a data space X, it has been shown [6] that $q_i(n)$ will converge to the maximal eigenvector of X, namely the first principal component of X (the vector describing an axis through the data, along which variance is maximized). In addition, $y(n)$ converges to the largest eigenvalue λ_1 of X (i.e. the expected value of variance along the q_i axis).

3. SOHVM Model Formulation

Let $X=[x_1, x_2, \ldots, x_{Ns}]$ be space of Ns data samples to be clustered, with each $x_i \in \Re^F$ representing an individual input sample vector drawn from X, then presented to the network at iteration i. Now let $W=[w_1, w_2, \ldots, w_{Nc}]$ represent the set of Nc cluster centres found in X so far, where $w_k \in \Re^F$. Similarly, let

$\Lambda=[\lambda_1, \lambda_2, \ldots, \lambda_{Nc}]$ and $Q=[q_1, q_2, \ldots, q_{Nc}]$ represent the set of maximal eigenvalue/eigenvector pairs (λ_k, q_k) describing the maximal variance of data in the vicinity of each associated cluster centre w_k (where $\lambda_k \in \Re$ and $q_k \in \Re^F$). In this way, each cluster prototype is essentially described by a dual memory element $v_k=\{w_k,(\lambda_k,q_k)\}$, capturing both cluster position and local variance.

The SOHVM process generates a topologically aware representation (map) $M=\{V,E\}$ of the dominant clusters existing in the data space X. The map consists of a set of prototype memory elements (*nodes*) $V=\{v_1, v_2, \ldots, v_{Nc}\}$, and a set of *edges* $E=\{e_1, e_2, \ldots, e_M\}$, where $e_m=\{v_l,v_p\}$: $v_l,v_p \in V$; $l \neq p$. The algorithm is shown in Figure 2, in summarized form.

3.1. Edge Formation and Topology

Edges e_k are formed in M via an edge ageing scheme (CHL), and can only form once $|E|>1$. With each x_i presented to the system, the two closest nodes to x_i are selected and correlated by forming an edge of $age=0$ between the two nodes (if one doesn't already exist). If an edge already exists, then its *age* is reset to zero. With the *winner* selected, all edges emanating from this node are aged by incrementing their respective *age* parameters. After each presentation is complete, edges in the network exceeding $Age_{max} \propto N_s$ ($Age_{max}<Ns$) are removed, leaving a skeleton of relevant associations.

3.2. Node Adaptation (ADAPT)

Assuming the winner has been chosen as the best representative x_i, and node generation is deemed unnecessary, information is then imparted to the network by updating the winner's centre w_{k*}, and its topological neighbours $w_{\eta(k*)}$ according to:

$$w_{k^*}(n+1) = w_{k^*}(n) + \alpha [\ x_i - w_{k^*}(n)\ ] \quad (4)$$

$$w_{\eta(k^*)}(n+1) = w_{\eta(k^*)}(n) + \alpha_{\eta b} [\ x_i - w_{\eta(k^*)}(n)\ ] \quad (5)$$

where k^* is the index of the winning prototype, and $0 \leq \alpha_{\eta b} << \alpha \leq 1$ are neighbourhood and winning node learning rates respectively. α decays exponentially with each iteration from an initial value (~0.1) and is reset as new nodes are spawned to foster local plasticity in the map. Associative learning ($\alpha_{\eta b}$) is fixed at a relatively low rate (~0.0005).

To adapt the winning node's eigenfilter, the co-ordinate system is then translated such that variance calculations are with respect to the prototype's centre. Specifically, an effective version of the input x_i' is found and used to train the variance probe (λ_{k*}, q_{k*}):

$$x_i' = x_i - w_{k^*}(n)\ ; \qquad y_i' = \sum_{j=1}^{F} x_{ij}' q_{k^* j}(n) \quad (6)$$

$$q_{k^*}(n+1) = q_{k^*}(n) + \alpha \cdot y_i' \cdot [\ x_i' - y_i' \cdot q_{k^*}(n)\ ] \quad (7)$$

$$\lambda_{k^*}(n+1) = \lambda_{k^*}(n) + \alpha \cdot [\ y_i' - \lambda_{k^*}(n)\ ] \quad (8)$$

3.3. Node Generation (GROW)

Essentially, the goal of node insertion is to bias the possible choices of inserting new nodes, in favour of locations along maximal points of variance with respect to the given winning node. In this sense, the chance of a node being generated on a noise point or outlier, and staying there, is reduced. Since we know that there must be *some* dense structures existing *within* the local scope defined by the maximal variance probe, then the assumption is that density outside this scope has been captured by a previously dispatched node. Nodes are ultimately inserted if they fall into the decision region $P \cap Q$ (Figure 1). The test for region Q offers direction selectivity, whilst P essentially represents an *annulus of significant similarity*: $(1/\beta)\cdot\sqrt{H(t)} < |x_i'| < \beta\cdot\sqrt{(\lambda_{k*})}$; where $\beta \geq 1$ represents an upper limit on the number of standard deviations from w_{k*} within which to trigger the spawning of a node. The network then enters a locating phase, allowing nodes to once again settle. $H(t)$ then decays: $H(t)_{new}=\max(\lambda_k)$; $\forall k$. β assists when there is only one node in the map, as the map grows, this decision region becomes more dependant on the difference between the local and maximal variances in the network. Continuation occurs if no new nodes form in an epoch, the network is not in a locating phase, and the network's nodes do not change significantly.

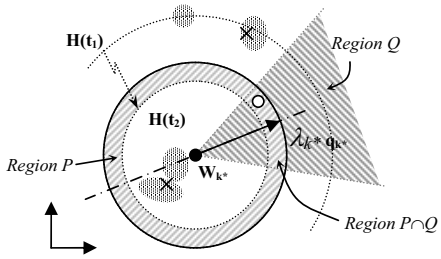

Figure 1: current prototype ● selects input ○ from region P∩Q for node insertion denying invalid inputs: **X**. Decision is thus more informed since it is based on the local variance probe (λ_{k*}, q_{k*}).

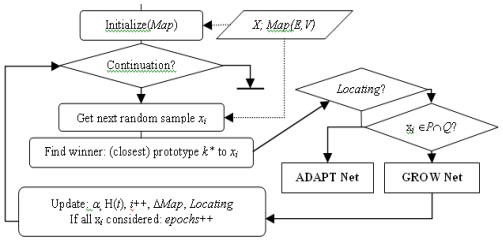

Figure 2: SOHVM algorithm

4. Results & Discussion

A number of sample datasets were generated for testing. Figures 3(a)(b) show clustering and evolution of both the GNG and SOHVM for two such datasets. Set A: 3x3 similar sized clusters, symmetrically spaced with differing densities; and set B: 9 randomly placed clusters, of different size and variance. In figures 3-4, prototypes are numbered according to order of discovery; associative links are in red (topology); and the blue lines represent major class boundaries.

The GNG works by accumulating error values between a winning prototype and inputs. At regular intervals, nodes are inserted along CHL formed edges connecting nodes with maximum accumulated error. Upon insertion, GNG redistributes accumulated error between the nodes split and the new node. From figures 2 & 3, it is apparent that GNG operates from the inside, out: a skeletal backbone grows from within, gradually attempting to span the data space. Regular insertion of nodes tends to lead to proliferation of prototypes, even with a possible error limit as a stopping criterion. An appropriate error value for a given data set must be known *a-priori*, or must account for all the sized clusters in the data, otherwise some dense clusters will be forced to sub-divide prior to all clusters being found. Figure 4 shows later phases of the GNG as it finally discovers remaining clusters – at which time previous clusters have been compromised.

The SOHVM quickly spans the entire dataset (favours insertion at ends of the maximal variance axis). Thus clusters most distant from one another are discovered earlier. This gives a more hierarchical breakdown of clusters: thus at any stage in its evolution, gives a representation of all the data to some degree of resolution. Most importantly, the SOHVM automatically decides on the *number* of clusters. Once all clusters are found, new inputs that are significantly dissimilar are likely to be outliers or noise (high variance and low density). Interestingly, this was found to drive H(*t*) high negating $P \cap Q$ and preventing further node production.

5. Conclusions

The framework and justification for a novel clustering technique (SOHVM) based on local variance driven self-organization was introduced. Compared with GNG, SOHVM has the ability to self determine an appropriate number of clusters within an unknown data set, and formulate a consistent set of prototypes without any *a-priori* knowledge. The approach shows promise even in difficult environments where the cluster sizes and densities may differ.

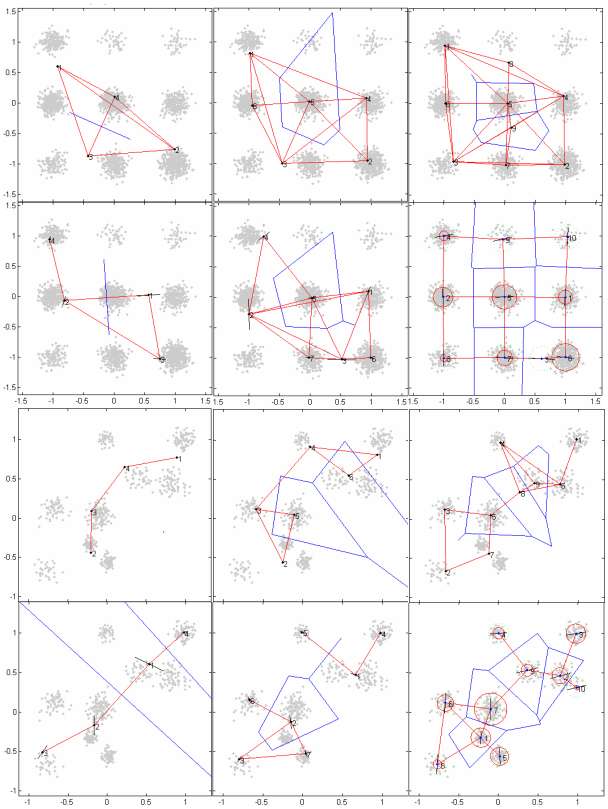

Figure 3: Clustering results for GNG (above) vs SOHVM (below); (a) Top: set A, (b) Bottom: set B; Evolution is from left to right. Rightmost SOHVM result is also the final result at convergence.

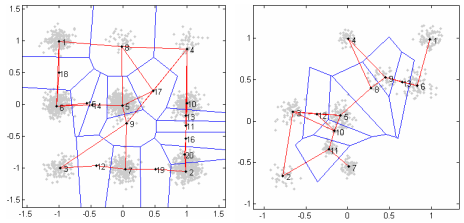

Figure 4: Error limited results for GNG

6. References

[1] A.K. Jain, R.P.W. Duin and J. Mao, "Statistical Pattern Recognition: A Review", *IEEE Transactions on PAMI*, 2000, Vol 22(1), pp. 4-37

[2] M.J. Kyan, L. Guan and S. Liss, "Refining competition in the self-organising tree map for unsupervised biofilm segmentation", *Neural Networks*, 2005 Vol 18(5-6), pp 850-860

[3] J. Pakkanen, J. Iivarinen and E. Oja, "The Evolving Tree, a hierarchical tool for unsupervised data analysis", *Proc. Int. Joint Conf. on Neural Networks*, Montreal, Jul31-Aug4, 2005, pp1395-1399

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

[5] T.M. Martinetz, "Competitive Hebbian Learning Rule Forms Perfectly Topology Preserving Maps" *In Proc. of Int. Conf. on Artificial Neural Networks (ICANN),* Amsterdam 1993 427-434.

[6] E. Oja, "A simplified neuron model as a principal component analyzer", *Journal of Mathematical Biology*, Springer, vol.15, pp. 267-273