2006 International Joint Conference on Neural Networks
Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

The Self-Organising Hierarchical Variance Map

Matthew J. Kyan, *Student Member, IEEE* and Ling Guan, *Senior Member, IEEE*

Abstract— **The Self-Organising Hierarchical Variance Map (SOHVM), a novel unsupervised clustering technique is proposed. Based on both Kohonen and Hebbian principles of self-organisation, the algorithm works to dynamically construct a topology preserved mapping of dominant prototype clusters from within an unknown data source. Each neuron in the network consists of a dual memory element that tracks information regarding a discovered prototype. In addition to position, Hebbian based Maximum Eigenfilters (HME) simultaneously estimate the maximal variance of local data. Competitive Hebbian Learning (CHL) is used to dynamically associate prototypes such that an accurate topology is maintained throughout the discovery process. Knowledge may then be progressively imparted to the network through appropriate neighbouring memory elements. Vigilance is assessed via interplay between local variances such that more informed decisions control and naturally limit network growth. The approach is closely related to Self-Organizing Tree Maps (SOTM), Growing Neural Gas (GNG) and their variants.**

I. Introduction

Unsupervised learning approaches, in attempting to describe an unknown set of data, find much application across a wide range of industries, particularly in bioinformatics, image processing and computer vision, as well as other applications that warrant significant need for data mining. The outcome of this mining can be used in such tasks as visualizing multivariate data of high dimension, modeling and formation of hypotheses regarding the nature of a set of data obtained under certain conditions, or in the testing and confirmation of hypotheses.

Computational technologies based on Artificial Neural Networks have been the focus for much research into the problem of unsupervised learning: in particular, network architectures based on principles of Self-Organisation. Such principles are in many ways, centred on Turing's initial observation in 1952 [1], namely, that *Global order can arise from Local interactions*. Receiving much neurobiological support, such mechanisms are believed to be fundamental to the organisation that takes place in the human brain.

Static methods such as Kohonen's self organising map (SOM), and non-neural based approaches such as K-means, and fuzzy based techniques have in many ways, formed a solid foundation for data mining research.

Self organising models such as SOM are consistently of particular interest in the research community. The reason for their popularity arises out of their ability to infer an *ordered* or *topologically preserved* mapping of the underlying data space. Relationships between discovered prototypes are captured through an imposed topology, i.e. an output layer connected in some predefined manner, such as a rectangular/hexagonal lattice. Mappings onto this layer (generally from some higher dimensional input space), are such that order is maintained: patterns near to one another in the input space map to nearby locations in the output layer.

Such mechanisms are useful for qualitative visualization [2,3], of high dimensional, multivariate data, where users are left to perceive possible clustered regions in a dimension that is more familiar to them (2D or 3D).

One of the most challenging tasks then, in any unsupervised learning problem arises by virtue of the fact that an attempt is being made to quantitatively *discover* a set of *dominant* patterns, clusters or classes in which to categorize underlying data without any knowledge of what an appropriate number of classes might be.

There are essentially two approaches taken as a consequence: either attempt to perform a series of separate clustering *runs* of an algorithm, over a range of different class numbers – and assess which yields a *better* result. Or, maintain a dynamic architecture that attempts to progressively realize an appropriate number of classes throughout the course of parsing a data set.

The latter of the two approaches is advantageous from a resource and time-of-execution point of view. As such, this is where the current presentation is focused. Many dynamic architectures have been proposed: particularly with regards to neural network approaches, as they seem particularly suited to developing a model of an input space, one item of data at a time: they evolve internally, through the progressive stimulation of individual samples.

Many techniques attempt to foster the extraction of hierarchical relationships within the data: groupings based on differing levels of granularity within the data to be clustered. Such techniques include the *Growing Hierarchical SOM* (GHSOM) [4-6], where an adaptive SOM lattice is grown until a map quality criterion is satisfied. After this, SOM's are branched according to quality measures achieved at higher levels, where the process is continued. Growth thus proceeds in a layered fashion.

Alternative, less rigid models such as the *Tree-Structured SOM* [7] attempt to partition a data space by building a tree hierarchy whilst parsing the data. The hierarchy formed acts

This work was supported by the Canada Research Chair program.

M.J.Kyan is with the School of Electrical and Information Engineering, University of Sydney, NSW, 2006 Australia (corresponding author: +1-416-9795000; fax: +1-416-9795000; e-mail: mkyan@ee.usyd.edu.au).

L.Guan is with the Department of Electrical and Computer Engineering, at Ryerson University, Toronto, ON, MK32E8 Canada (e-mail: lguan@ee.ryerson.ca).

0-7803-9490-9/06/$20.00/©2006 IEEE

as a mechanism for efficient fast winner search. Similar principles are adopted in the recent Evolving Tree algorithm [8], wherein the frequency with which a given node is fired accumulates, acting as an indicator for subdivision of the network. Subdivision is thus controlled by a threshold, which decays over time to hinder the growth of the tree. New nodes (a predefined number) then form children at the site of the original winner, which is subsequently retired from competition and retained in memory for use in a top down search for future winning nodes.

One of the problems with such approaches, however, is that nodes from differing branches lower in the hierarchy may in fact move closer to one another during later iterations. In doing so, higher resolution clusters targeted in the search, may lead to errors in matching winners with the current input. Subdividing the input space along with the tree network to some degree resolves the search dilemma, however, errors in coarse segmentations at higher levels of the hierarchy, may result in parts of natural clusters in the data at higher resolutions being disassociated from one another within the hierarchy. Clearly such approaches need a mechanism to redistribute data (and/or nodes) between sub-trees within the overall hierarchy as needed.

In this paper, we attempt to address some of the aforementioned issues with a novel model based on hierarchical vigilance assessment and local maximal variance estimation. The proposed network is called the Self-Organising Hierarchical Variance Map (SOHVM). The network aims to insert new nodes in regions of maximal variance with respect to the nodes currently representing the data space. A combination of a globally decaying vigilance gives preference to samples drawn from the bounding regions of a data space whilst generating a network of prototypes. Simultaneous variance assessment updates the network's notion of how the dataset is bounded. As nodes are inserted into the map, each new unit maintains its own assessment of the way local data is bounded, and so forth. The result is a hierarchical assessment of the data space that provides an efficient mechanism by which the search for new clusters may be constrained, and the insertion of new nodes may be more informed.

II. Background and Related Work

A. Competitive Hebbian Learning

Inspired by Hebb's postulate of learning [9], the principal of *Competitive Hebbian Learning* (CHL) [10] was first introduced to more accurately estimate the underlying topology of an input space.

CHL offers a means of strengthening and weakening associations between discovered prototypes in dynamically generating self-organising architectures. This is achieved by forming connections between pairs of nodes that are found to be most representative of any given input: akin to triggering or refreshing a correlation between the two nodes. CHL discovers an *induced voronoi triangulation* [10] from a partitioned data space: a triangulation across the set of cluster prototypes that is restricted to the dense regions. Thus topology is detected and constructed dynamically.

CHL is usually implemented by an edge ageing scheme. Stimulation of the two closest nodes to an input gives *birth* to an edge, which is progressively weakened unless re-stimulated by other inputs from the underlying density. Edges re-stimulated are reset to a zero age, whilst old edges are eventually removed from the network, thereby disassociating prototypes.

B. Growing Neural Gas

Topological information mined through CHL has found its way into many self-generating architectures, such as the Growing Neural Gas (GNG) algorithm [11], and variants: Growing Cell Structures (GCS) [12], Growing Grid (GG) [13] and DASH [14]. The GNG works by accumulating error values between a winning prototype and inputs. At regular intervals, new nodes are inserted along CHL formed edges at sites of maximally accumulated error. A node with the largest error is selected, along with its most error prone neighbour. The edge connecting these two nodes is split by inserting a new node at a location midway. This process attempts to efficiently allocate prototypes to relevant regions of density and continues until a maximum error criterion is satisfied by all nodes within the network.

Due to the online nature of measuring error statistics, these methods simulate *average* distortion error within each node, by accumulating, then redistributing error between the original error prone nodes, and their inserted *offspring*. Error values throughout the network are decayed over time to emphasize the accumulation of more recent errors over those accumulated due to past network states.

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 *apriori*, or must account for all the sized clusters in the data, otherwise dense clusters may possibly be forced to sub-divide prior to all clusters being found. Distortion is extremely sensitive to the selection of the period (λ) over which error statistics and node insertions are evaluated.

C. Self-Organising Tree Maps

The *Self-Organising Tree Map* SOTM was first proposed in [15], and later developed [16,17]. Like GNG, the structure of the network is dynamic, and is grown from a single node, however topological relationships are formed in a hierarchical manner.

The SOTM is somewhat of a hybrid between the traditional Kohonen SOM and Adaptive Resonance Theory (ART) architectures [18-20]. Like ART, a set of prototype nodes are gradually formed as the input space is parsed. This process is overseen by a *vigilance* test. If the input is within a tolerance of the winning prototype, *resonance* is said to occur, resulting in the refinement of the winning prototype.

Otherwise, whilst there is still capacity, a new prototype is formed. Like SOM however, prototypes so discovered also contain interconnections that allow for the parsing process to be more ordered or *topologically* aware.

Node insertion in the SOTM is thus driven by a top-down process that effectively explores the data space from the *outside-in*. An ellipsoid of *significant similarity* [17], 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. Samples encountered outside the scope of the current network's prototypes will result in the generation of new nodes in the network model. Thus new prototypes form as children of their associated winning parent nodes, resulting in the progressive formation of 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. This mechanism acts as a form of hierarchical control that favours partitioning of low resolution clusters in early phases of learning, whilst favouring higher resolution partitioning later.

Due to the random nature in which samples are presented to the network during training, the SOTM does suffer from some limitations. Node generation, for instance, 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 always maintained in latter stages (lower levels of the tree). The newly proposed SOHVM attempts to address these issues.

III. The SOHVM Framework

A. SOHVM Model Formulation

The SOHVM process is formulated as follows. Firstly, let $X=[x_1, x_2, \ldots, x_{N_x}]$ represent an input data space to be clustered (with a total of N_x samples). Each $x_i \in \Re^F$ represents an individual input pattern vector drawn from X, and presented to the network at iteration i.

The SOHVM process then works to generate a topologically aware representation (map) $M=\{V,E\}$ of the dominant clusters existing in the data space X. The map is formed as a network consisting of a set of prototype memory elements (*nodes*) $V=\{v_1, v_2, \ldots, v_{N_c}\}$, each connected by a set of *edges* $E=\{e_1, e_2, \ldots, e_{N_e}\}$, where $e_m=\{v_l, v_p\}$: $v_l, v_p \in V$; $l \neq p$. Where N_c and N_e represent the number of classes/clusters and the number of edges respectively. Each node v_k in M contains a dual memory element $v_k=\{w_k,(\lambda_k,q_k)\}$: where $w_k \in \Re^F$ represents the prototype's position (cluster centre), while (λ_k,q_k): $\lambda_k \in \Re$, $q_k \in \Re^F$ forms a maximal eigenvalue/eigenvector pair describing the maximum variance of data in the vicinity of the k^{th} cluster. This second memory element acts as a *local variance probe*. These probes are then used as an aide in controlling the growth of the network.

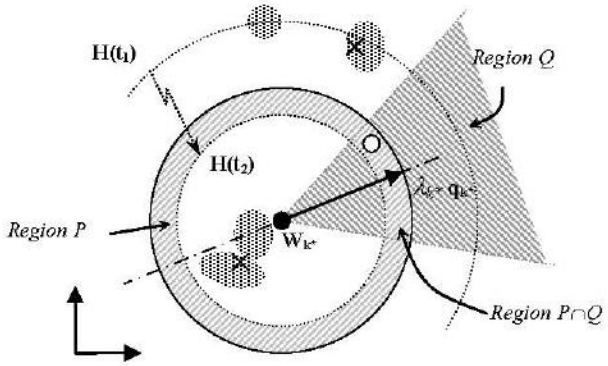


Figure 1: current prototype ● accepts input ○ from region $P \cap Q$ for node insertion denying invalid inputs: X. $H(t)$ falls below the maximal variance as described by the eigenvector (λ_{k^*},q_{k^*}), promoting a subsequent hotspot region $(P \cap Q)$ within which an encountered input sample will spawn a new node: insertion is thus more informed than in the SOTM.

B. Top-down and Bottom-up Vigilance Testing

The driving force behind the proposed SOHVM model revolves around *how* the decision is made regarding whether or not to grow or adapt the network.

In the SOTM, simply exceeding a globally decaying threshold $H(t)$ is enough to inspire growth in the map. Proximity measures alone, however, do not give much credence to what might be the most *likely* regions of input space that *warrant* further exploration or modeling. Instead, the SOHVM attempts to make a more *informed* decision for node insertion by consideration of both a *top-down* global threshold and localized *bottom-up* variance information.

Bottom-up information is extracted by a *Hebbian-based maximum eigenfilter* (HME) which implements the local variance probe mentioned earlier. Alongside w_k, which captures cluster position, the HME (λ_k,q_k) simultaneously estimates the nature of the data in the vicinity of each prototype. The HME principle is described in section III-C. Interplay between such probes and the top-down parsing mechanism $H(t)$ allows for a semi-informed decision region $P \cap Q$ (see Figure 1) for triggering network growth. Samples falling within this scope of an existing winning node (v_{k^*}) result in the formation of a new prototype (outlined in section IV-B). Input samples that do not form new prototypes instead cause the existing set of prototypes to adapt via competitive means (outlined in section IV-D).

C. Hebbian-based Maximum Eigenfilter

The HME [21] offers a useful mechanism that fits well into the online framework of our proposed self-organising architecture. 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:

$$y = \sum_{j=1}^{N_I} q_{ij} x_i' \quad (1)$$

$$q_i(n+1) = q_i(n) + \eta \cdot y(n) q_i(n) \quad (2)$$

$$q_i(n+1) = q_i(n) + \eta \cdot y(n)[\ x_i'(n) - y(n) q_i(n)\ ] \quad (3)$$

where x is the input vector, y is the output of the perceptron, and η is a learning rate, and $q_i(n)$ is a synaptic weight vector.

As the perceptron is trained with samples from a data space X', it has been shown [21] 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, $y(n)$ will simultaneously converge to the largest eigenvalue λ_1 of X' (i.e. the expected value of the variance along the axis defined by q_i).

It is important to note that to perform such calculations within the *local* context of an individual node, vectors in the original data space must undergo a co-ordinate transformation so that any adaptation or growth steps are performed with respect to the centre of the winning node (i.e. w_{k^*}).

D. Growth Trigger & Adaptation

The possible choices for inserting new nodes are biased to favour 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. By refashioning the decision process from that of the SOTM, we delay even making a growth decision, until such time as the global hierarchical function $H(t)$ drops to within the variance scope of any given winning node.

$H(t)$ is initialised to a value that exceeds the maximal range in the data: say four times the standard deviation of the input data. There it stays until the map stabilizes. At this point, the node with the highest degree of variance as probed by existing nodes is located (at the outset, this will be due to the initial node in the map). Local variance indicates that *some* dense structures must exist *within* the local scope defined by each maximal variance probe. We then step $H(t)$ down from this global scope, to the maximum variance probed by any prototype thus far. The jump down in $H(t)$ allows the network to evaluate a specific, focused region of the data space, about each of its current nodes.

Nodes are ultimately inserted if they fall into a decision region $P \cap Q$ which forms in the more variant regions in the data (Figure 1). The assumption is then, that density outside this scope has been captured by a previously dispatched node, so the growth process should work to progressively partition these dense regions.

As the dense regions are partitioned, further stimulation from randomly sampled inputs (not resulting in node insertion), cause the existing prototypes to adapt, and refine their locations to better represent and converge upon clusters present in the underlying data.

IV. The SOHVM Algorithm

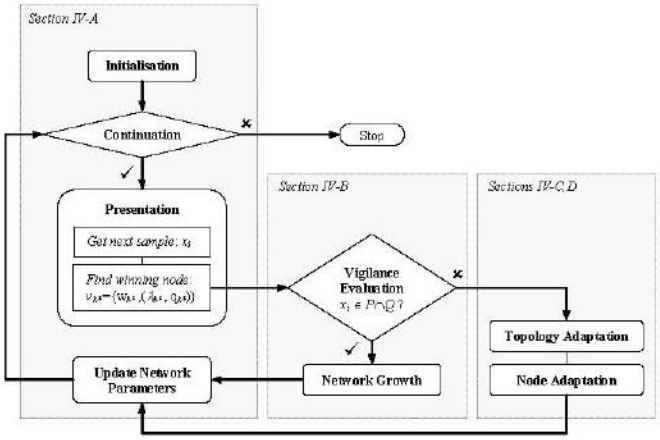


Figure 2: The simplified SOHVM algorithm.

The algorithm for generating the map is summarized in a simplified flowchart depicted in Figure 2. Details associated with each of the main components are given in the following sections.

A. Initialisation, Continuation and Presentation

The process of *initialisation* begins by forming a map (M) with a single node only. The initial (root) node v_0 is a randomly selected sample x_{t_0} from X. That is to say, $w_0 = x_{t_0}$. The variance probe for the root node is then initialised to a unit eigenvector of random orientation and zero eigenvalue.

The algorithm then proceeds through a continuation test before sampling inputs randomly, presenting and evaluating each against the current state of M.

Presentation of each individual input involves first identifying the best matching unit (BMU) or winning prototype from within the current map. This is achieved by selecting the closest prototype according to some distance/similarity metric v_{k^*}: $k^* = argmin_j d_j(x_t, w_j)$; (e.g. d_j might be Euclidean, as used in this work). As such, all existing prototypes essentially compete for the opportunity to learn something from the input pattern, and the closest prototype (indexed by k^*), is regarded as the winning node. The current input is either used to trigger map growth, or adapt the current map prototypes until some stop criterion is met.

The criterion for *continuation* is generally a combined condition requiring that no new nodes have formed within an epoch, and no significant changes have occurred in the map's prototypes. After performing any necessary growth or adaptation steps (sections IV-B,C,D), the network parameters are *updated* before iterating back through the continuation test to consider the next sample, where the process repeats. Network parameters include incrementing counters (epochs, iterations), decrementing learning rates ($\alpha(t)$ - detailed in section IV-D), and calculating any changes in the map ($\Delta M_i = |M_i - M_{i-1}|$; if $|M_i| \neq |M_{i-1}|$, then $\Delta M_i = \infty$).

B. *Vigilance Evaluation and Network Growth*

As noted in Figure 1, the current input is then tested against the semi-informed decision region $P \cap Q$, to determine whether or not to grow or adapt the map.

Region P scribes an *annulus of significant similarity* within the input space, emanating from the winning node such that $(1/\beta)\cdot\sqrt{H(t)} < |\, x_i' \,| < \beta\cdot\sqrt{(\lambda_{k^*})}$; 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. This parameter is used to avoid the situation in which there is only a single node in the network. In this situation, if $H(t)$ was to step down to the maximum variance discovered, then $H(t)=\lambda_{k^*}$ effectively shutting off the decision region. Empirically, we found $1.1 \leq \beta \leq 1.3$ to be a reasonable range that yielded consistent results across a broad range of data distributions.

The test for region Q makes use of the orientation selectivity inherent in the HME, focusing this bias further. This is normally achieved by evaluating the inner product: $[x_i' \cdot q_k]$ (e.g. an inner product of $\geq 1/\sqrt{2}$ would restrict the region $P \cap Q$ to a sector between ±45°).

A new node is generated and initialized to the position at which such a valid input sample is encountered. The network then enters a *locating* phase, allowing nodes to adapt and re-stabilise, before growth consideration is resumed.

Adaptation is broken into two parts – both for updating edges to reflect topological status of the map, and node prototype information (sections IV-C and IV-D).

C. *Topology Adaptation*

An essential step in adapting prototypes is the estimation of underlying topology. In the SOHVM, this is achieved through a combination of SOTM tree links, and an adaptive CHL mechanism. The SOTM process of inserting a new node and attaching it to the winning node tends to form a tree structure as depicted earlier. Over time however, such a tree may lose significance in lower branches: clusters may have been disassociated before enough information was known. The SOHVM attempts to maintain the tree hierarchy early, but allows this condition to be relaxed as new associations take shape later in the evolution of the network. The hierarchical structure is important during evolution as it helps the SOHVM to efficiently span the data across all phases of network resolution.

Edges e_k are either inserted during growth, or formed in M via an edge ageing scheme similar to GNG, and can only form once $|V|>1$. 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 above a certain limit Age_{max} are removed, leaving a skeleton of only the most relevant associations currently in the network.

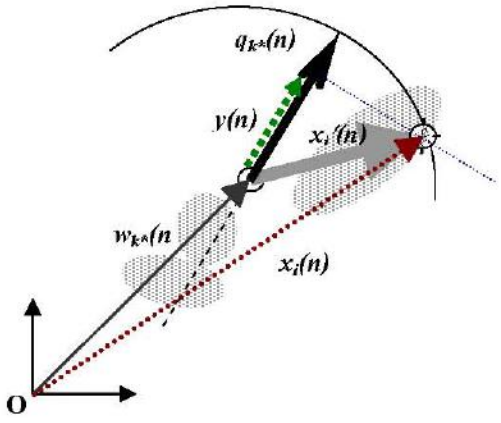


Figure 3: Recasting the calculation of maximal variance such that it relates to the scope of an individual node in the network. The graphic is a geometric representation of equations (6)-(8).

D. *Node Adaptation*

Once the winner has been chosen as the best representative x_i, and node generation is deemed unnecessary by the vigilance test, information is then imparted to the network by updating both the winning node $v_{k^*}=\{w_{k^*},(\lambda_{k^*},q_{k^*})\}$, and selected neighbours, as indicated by the adapting topology. This involves two major steps: updating the node's position w_{k^*}, and secondly, the node's variance probe (λ_{k^*},q_{k^*}).

The winner's centre w_{k^*}, and its topological neighbours $w_{\eta(k^*)}$ are updated according to:

$$w_{k^*}(n+1) = w_{k^*}(n) + \alpha(t)\left[\, x_i - w_{k^*}(n) \,\right] \tag{4}$$

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

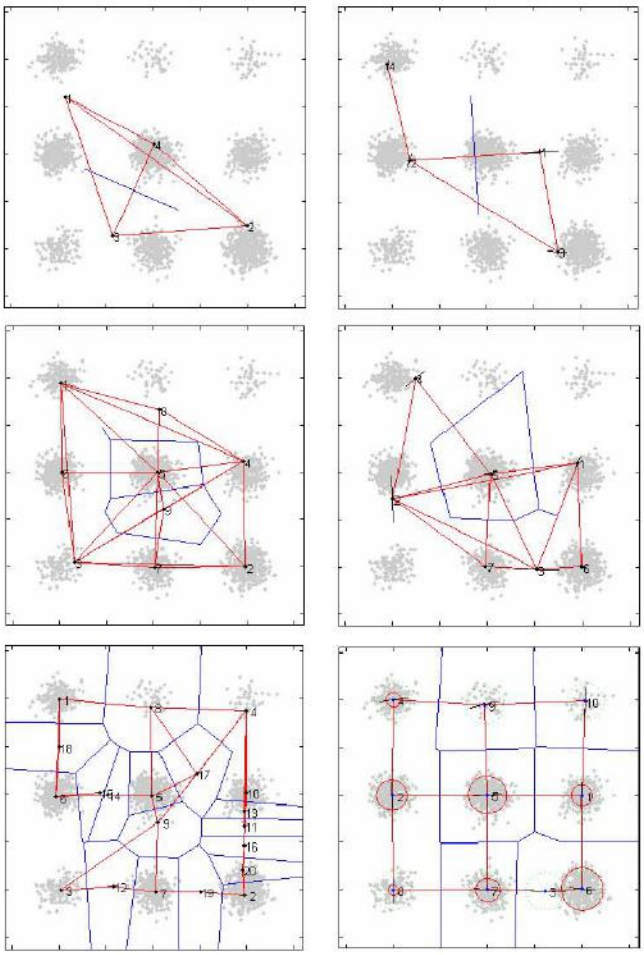
where k^* is the index of the winning prototype, and $0 \leq \alpha_{\eta b} << \alpha(t) \leq 1$ are neighbourhood and winning node learning rates respectively. $\alpha(t)$ decays exponentially with each iteration from an initial value (~0.1) and is reset as new nodes are spawned to foster local plasticity 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 in equations (1)-(3) are now made with respect to the prototype's centre as in equations (6)-(8). This transformation is outlined in Figure 3, where x_i' represents an *effective* version of the input pattern x_i. This effective version of the input is found and used to train the variance probe (λ_{k^*},q_{k^*}). Training occurs after casting the Hebbian unit into the reference frame of the winning node's centre, and thus proceeds through:

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

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

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



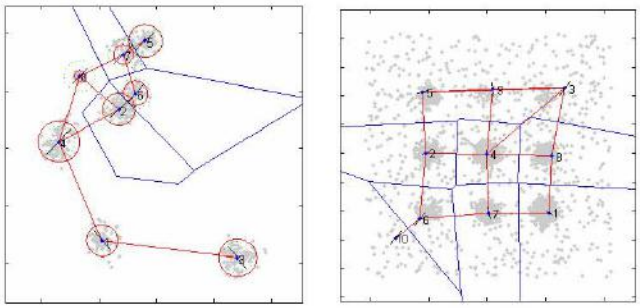
(a) (b)

Figure 4: (a) GNG evolves out onto data space, but splits prematurely well before capturing all cluster positions. Classification thus subdivides some natural clusters. (b) SOHVM maximally spans the data space through hierarchical partitioning: this property is localised through the HME units (dark lines show axes for orientation selectivity). Network infers more accurate number of total clusters. SOHVM also forms a more faithful or intuitive topological representation of the data at all resolutions.

V. Results and Discussion

Simulations were conducted on a number of synthetic datasets, each consisting of a series of clusters in 2D input space, of varying densities, size and separation. Comparisons were drawn between both dynamic and static unsupervised architectures.

In the first comparison, performance of the SOHVM against GNG is demonstrated (Figure 4). In this figure, a set of equally sized clusters was arranged in a 3x3 grid across the data space. The clusters were symmetrically spaced yet of varying densities; Prototypes are displayed on the map as nodes, numbered according to order of discovery. Associative links (topological edges) connect these nodes (in red), whilst blue lines represent major voronoi class boundaries.



(a) (b)

Figure 5: Sample clustering on a different 2D data sets (a) using variously sized, spaced and density clusters. In (b) top right, data from Figure 4 is contaminated with noise: outlier triggers stop. Connections between nodes represent final associations.

From Figure 4(a), 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 *apriori*, 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. Later phases of the GNG show that by the time all clusters are finally discovered, previous clusters have been compromised.

In contrast, the SOHVM quickly spans the entire dataset (by favouring 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, and consequently, at all stages in its evolution, gives a more faithful representation of the underlying topology of the input space. Most importantly, the SOHVM automatically decides on the final number of clusters. A likely reason for this is that 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 with respect to variance probes, negating the *annulus of significant similarity* and preventing further node production. A similar situation occurs in the noise contaminated version of this data set (Figure 5b).

In Figure 6, preliminary comparisons are conducted between SOHVM and some popular static methods – Kmeans [22], Fuzzy C-Means (FCM) [23], and advanced FCM variants: Gustafson-Kessel (GK) [24] and Gath-Geva (GG) [25]. Whilst the SOHVM was allowed to automatically infer an appropriate number of clusters, the static methods were passed this same number as a parameter. The goal here was to gain some level of insight into the efficiency of allocation (cluster representation) offered through the SOHVM against methods even when the user supplies such *apriori* information regarding the desired number of clusters in the data.

Density contours were projected over the cluster results using tools from [26]. As seen in Figure 6, SOHVM finds a

more reasonable set of cluster positions. Fuzzy methods such as GK and GG do demonstrate some propensity toward capturing elongated and different shaped regions, and show sensitivity to a possible (perceived) overlapping cluster in the mid top right. This is however, at the expense of intuitively obvious clusters in the mid lower region. Furthermore, GK and GG operate following an FCM initialization, yet still do not resolve two clusters in the lower central region. FCM and K-means on the other hand remain quite sensitive to initialization of cluster centres, and it is often left to the user to decide when an appropriate solution is found.

VI. 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 *apriori* knowledge. The approach shows promise in difficult environments where the cluster sizes and densities may differ, or noise is present. With respect to static unsupervised fuzzy techniques that are initialised with the knowledge of an appropriate number of clusters to look for, the SOHVM still appears to locate clusters in a more efficient manner (albeit automatically). SOHVM is a technique that offers a more objective breakdown of an input space into an appropriate number of dominant patterns which may be used in the description of an otherwise unknown data source. Continued testing should lead to this technique finding application in a number of pertinent real world problems wherein the modeling of unknown complex data sources may lead to the development and confirmation of hypotheses regarding the nature of the environments from which the data are extracted.

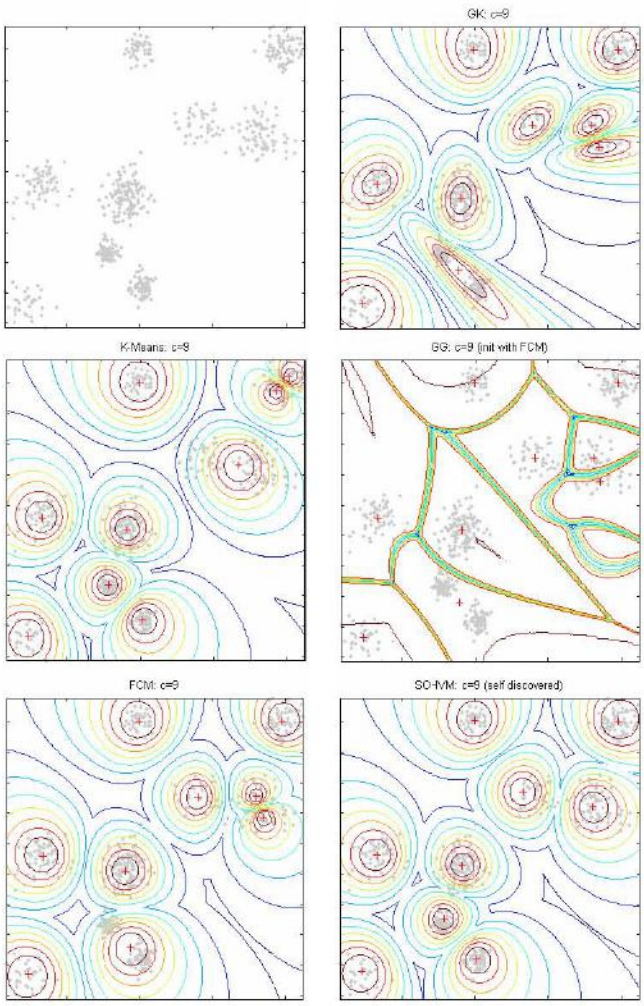


Figure 6: Comparison of SOHVM with popular static clustering methods. SOHVM demonstrates greater efficiency in allocating nodes to perceived clusters and does so without knowledge of an appropriate total number of classes into which it should cluster. The top left shows the original synthetic dataset, consisting of 9 clusters of unequal density and size. A total of 9 clusters was predefined for the other methods.

Acknowledgment

Matthew Kyan is supported by the Canada Research Program (CRC), and would sincerely like to thank Mr. Kambiz Jarrah for the many in depth discussions that have given inspiration to much of this work.

References

[1] A.M. Turing, "The chemical basis of morphogenesis", *Philosophical Transactions of the Royal Society, B*, 1952 vol. 237, pp. 5-72

[2] A. Ultsch, "Self-organizing neural networks for visualization and classification", In O.Opitz, B. Lausan, and R. Klar, eds., *Information and Classification*, pp. 307-313. Springer, Berlin, 1993.

[3] S. Kaski and T. Kohonen, "Exploratory data analysis by the self-organizing map: Structures of welfare and poverty in the world", In A.P.N. Refenes, Y. Abu-Mostafa, J.Moody, and A. Weigend, eds., *Neural Networks in Financial Engineering*, pp. 498-507. World Scientific, Singapore, 1996.

[4] M. Dittenbach, D. Merkl, and A. Rauber, "The growing hierarchical self organizing map," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN 2000)*, S. Amari, C. L. Giles, M. Gori, and V. Puri, Eds., vol. 6. Como, Italy: IEEE Computer Society, July 24–27 2000, pp. 15–19.

[5] M. Dittenbach, A. Rauber, and D. Merkl, "Recent advances with the growing hierarchical self-organizing map," in *Advances in Self-Organizing Maps*, N. Allinson, H. Yin, L. Allinson, and J. Slack, Eds. Lincoln, GB: Springer-Verlag, June 13-15 2001, pp. 140–145.

[6] M. Dittenbach, A. Rauber, and D. Merkl, "Uncovering the hierarchical structure in data using the growing hierarchical self-organizing map," *Neurocomputing*, vol. 48, no. 1–4, pp. 199–216, November 2002.

[7] P. Koikkalainen and E. Oja, "Self-organizing hierarchical feature maps," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN 1990)*, San Diego, CA 1990.

[8] 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

[9] D.O. Hebb, *The Organization of Behavior: A Neuropsychological Theory*, 1949, New York: Wiley.

[10] T.M. Martinez, "Competitive Hebbian learning rule forms perfectly topology preserving maps", In *Proceedings of the International Conference on Artificial Neural Networks (ICANN 1993)*, Amsterdam, Netherlands, pp. 427-434.

[11] 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. MIT Press, 1995, pp. 625–632.

[12] B. Fritzke, "Growing cell structures – A self-organizing network for unsupervised and supervised learning," *Neural Networks*, vol. 7, no. 9, pp. 1441–1460, 1994.
[13] B. Fritzke, "Growing Grid – A self-organizing network with constant neighborhood range and adaption strength," *Neural Processing Letters*, vol. 2, no. 5, 1995.
[14] C. Hung and S. Wermter, "A Dynamic Adaptive Self-Organising Hybrid Model for Text Clustering", *Third IEEE International Conference on Data Mining (ICDM 2003)*, pp. 75.
[15] H. Kong and L. Guan, "Detection and removal of impulse noise by a neural network guided adaptive median filter," *Proc. IEEE Int. Conf. on Neural Networks*, pp. 845-849, Perth, Australia, 1995.
[16] J. Randall, L. Guan, X. Zhang, and W. Li, "Investigations of the self-organizing tree map," *Proc. Of Int. Conf. on Neural Information Processing*, vol 2, pp. 724-728, November 1999.
[17] M. Kyan, L. Guan and S. Liss, "Refining competition in the self-organising tree map for unsupervised biofilm segmentation", Neural Networks, Elsevier, Jul-Aug. 2005 Vol 18(5-6), pp 850-860
[18] S. Grossberg, *Studies of Ming and Brain*, Boston: Reidel, 1982.
[19] G.Carpenter, and S. Grossberg, "the ART of adaptive pattern recognition by a self-organizing neural network", *Computer*, vol. 21, no. 3, pp.77-87, 1988.
[20] G.A. Carpenter, S. Grossberg, and J.H. Reynolds, "ARTMAP: supervised real-time learning and classification of nonstationary data by a self-organizing neural network", *Neural Networks*, vol. 4, pp. 565-588, 1991.
[21] E. Oja, "A simplified neuron model as a principal component analyzer", Journal of Mathematical Biology, Springer, vol.15, pp. 267-273
[22] C.W.Therrien, *Decision estimation and classification – An Introduction to Pattern Recognition and Related Topics*. John Wiley and Sons, New York, pp. 215-227, 1989
[23] J.C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, 1981.
[24] D.E. Gustafson and W.C. Kessel, "Fuzzy clustering with fuzzy covariance matrix", In *Proceedings of the IEEE CDC*, San Diego, pp. 761-766. 1979.
[25] I. Gath and A.B. Geva, "Unsupervised optimal fuzzy clustering", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7, pp.773-781, 1989.
[26] B. Balasko, J. Abonyi and B. Feil, *Fuzzy Clustering and Data Analysis Toolbox for use with Matlab*, Department of Process Engineering University of Veszprem, Hungary, http://www.fmt.vein.hu/softcomp/