The Architecture and Performance of a Stochastic Competitive Evolutionary Neural Tree Network

N.Davey, R.G.Adams, S.J.George

N.Davey@herts.ac.uk, R.G.Adams@herts.ac.uk, S.J.George@herts.ac.uk
Faculty of Engineering and Information Sciences,
University of Hertfordshire,
Hatfield, Herts., UK. AL10 9AB
Tel +44 01707 284321

Abstract: A new dynamic tree structured network - the Stochastic Competitive Evolutionary Neural Tree (SCENT) is introduced. The network is able to provide a hierarchical classification of unlabelled data sets. The main advantage that SCENT offers over other hierarchical competitive networks is its ability to self-determine the number and structure of the competitive nodes in the network without the need for externally set parameters. The network produces stable classificatory structures by halting its growth using locally calculated, stochastically controlled, heuristics. The performance of the network is analysed by comparing its results with that of a good non-hierarchical clusterer, and with three other hierarchical clusterers and its non stochastic predecessor. SCENT's classificatory capabilities are demonstrated by its ability to produce a representative hierarchical structure to classify a broad range of data sets.

Keywords: Dynamic neural tree, self organising, hierarchical classification, unsupervised learning, cluster analysis, neural network, competitive learning, inductive discovery, structured knowledge.

1. Introduction

In this paper we discuss a novel neural network model that is capable of discovering hierarchical structure and categories from unlabelled data.

Simple competitive neural networks are straightforward methods for finding structured knowledge, in that any inherent clustering in a set of data is discovered. The process of discovery is, however, limited to the discovery of a predefined number of clusters. A dynamic network is set a significantly more difficult problem: to find not only the clusters but also their number, by dynamically creating classifying neurons in response to apparent structure in the data. A still more difficult task in the discovery of structured knowledge is the problem of finding the hierarchical clustering structure in a data set. Here, natural clusters, their number, and their hierarchical relationship are sought. Moreover if such a hierarchy can be found then the codevectors of the network can be considered as prototypes for the categories they classify, see the discussion in Section 4.2

Competitive Learning Neural Networks (CLNNs) [Hertz et al., 1991] classify unlabelled data sets, providing single layer cluster analysis. The nature of the classification found by a CLNN is strongly influenced by the number of competitive nodes used, which must be decided upon before starting. As CLNNs produce single layer classifications they are not suitable for use in application areas such as biological taxonomy [Sneath and Sokal 1973] that require a hierarchical analysis of the data.

Recent research [Li et al., 1995, Racz and Klotz 1991, Butchart et. al. 1997, Adams et. al. 1998, Song and Lee, 1998] has investigated the possibility that the nodes could be arranged in a tree structure; such an arrangement has two potential benefits: searching a tree is fast and any hierarchical information in the data may be explicitly represented in the tree. Most clustering algorithms, neural net or otherwise, expect the number of classes used to be predefined; this is an obvious problem if no a-priori knowledge about the data is available. Consequently further research has been directed towards exploring the possibility that a network can dynamically evolve the appropriate structure in response to the data. Such an approach naturally leads to tree structures - a unit may produce child nodes if it feels it is not classifying its local data with sufficient granularity.

Dynamic Neural Tree Networks (DNTNs) [Li et. al., 1992, Racz and Klotz 1991] attempt to combine the advantages of using a tree structure and of growing the network, these networks are reviewed in Section 2. DNTNs show promise in their ability to produce stable yet flexible hierarchical structures. The nature of the hierarchical structure and the quality of the final classification produced by the networks is, however, very strongly influenced by the values given to externally set parameters. A new network, the Competitive Evolutionary Neural Tree (CENT) [Butchart 1996c], was developed to extend the ideas of DNTNs.

Most unsupervised neural networks perform some form of gradient descent on an error function, normally the sum squared error, and as such are likely to fall into a local minimum. Attempts to overcome this problem have concentrated on two approaches: modifying the objective function and thereby the update rule, and secondly using some form of simulated annealing [Metropolis et al., 1953]. In practice, in both approaches, all the nodes in the network are updated on each vector presentation, rather than just the winner. The term *soft competition* is used to embrace both methods.

One of the most effective models that uses soft competition in a non-hierarchical clusterer is the Neural Gas model [Martinez et al., 1993]. Here all nodes are updated at each iteration with the competition becoming progressively less soft over time, with the network eventually approaching the standard single winner CLNN paradigm. The success of the simulated annealing approach, as exemplified in Neural Gas, encouraged the exploration of mediated stochasticity within the context of CENT, which produced a more robust model Stochastic CENT, SCENT, introduced in Section 3.

For the purposes of the results reported here we have compared our model with the two other dynamic neural trees, the dynamic competitive learning technique of Racz and Klotz [Racz and Klotz, 1991], and the Neural Tree of Li [Li et al., 1992] and with the best of the non-hierarchical clusterers - the Neural Gas model. We have also compared the representational ability of our SCENT network with another hierarchical neural clusterer - the HART network of Bartfai [Bartfai, 1995]. Results of these comparisons can be found in Section 4.

2. Background

Cluster analysis is the production of a classification scheme for initially unclassified and unlabelled data; the aim being that the classification produced should represent the natural groups/clusters that may be present in the data.

Unsupervised clustering using neural networks is a well established technique, with the simple competitive learning neural network (CLNN) being the elementary model and Kohonen's self organising maps (SOMs) the most popular manifestation [Hertz et al., 1991].

In a simple competitive network the weight vectors of the nodes in the output layer of the network after training should ideally represent the prototypes of the clusters found in the data set. The initial values for these prototypes - the weight vectors - are normally set to random values in an attempt to avoid bias. Each input from the data set is then presented to the network in turn and output layer nodes compete for the right to classify the input and update their weight vectors. The winner of the competition is the node whose weight vector is most similar to the input vector presented. If the winning node's weight vector is $\mathbf{w}_{\mathbf{i}*}$ then:

$$|\mathbf{w}_{\mathbf{i}*} - \mathbf{x}| \le |\mathbf{w}_{\mathbf{i}} - \mathbf{x}| \quad \text{(for all i)} \qquad (1)$$

where:

$\mathbf{x}$ is the input vector and

$\mathbf{w}_{\mathbf{i}}$ are the prototype weight vectors

The winning node has its weight vector updated to make it more similar to the input that caused it to win according to:

$$\mathbf{w}_i(n) = \mathbf{w}_i(n-1) + \alpha[\mathbf{x}(n) - \mathbf{w}_i(n-1)] \qquad (2)$$

where:

α is the learning rate of network, normally in the range 0.01 to 0.15, and

n is the time step.

The output nodes gradually move towards the centre of the group of inputs which they classify, the node's weight vector becoming the prototype for the cluster of inputs it classifies. In this way the clusters that exist in a data set can be discovered. The network is iteratively reducing the Sum of Squared Error (SSE) by applying (2) to the winning node, where SSE is defined by:

$$\text{SSE} = \sum_{i=1}^{c} \sum_{\mathbf{x} \in X_i} |\mathbf{x} - \mathbf{m_i}|^2 \qquad (3)$$

where:

c is the number of clusters, $\mathbf{x}$ are the input vectors,
X_i is the ith cluster and $\mathbf{m_i}$ is the mean vector in X_i

The behaviour of the network is very similar to that of the statistical technique of k-means clustering. If the output nodes are updated in batch mode, that is updated once per presentation of the whole data set, and dead units (units that are initialised too far from the data and never win the competition) are prevented then the CLNN has been shown to be functionally equivalent to k-means clustering [Krishnaiah and Kanal, 1989].

In a SOM the classifying units are arranged with some spatial topology, usually a grid. The most common form of ordering for a SOM is a two dimensional lattice providing a two dimensional output space. The winning node for each input is found and updated in the same way as for a CLNN. However, in a SOM the neighbours of the winning node are also updated, where the neighbours of a node are defined by the ordering imposed on the output nodes. Typical neighbours for a two dimensional lattice are contained within a square grid centred on the winning unit. This neighborhood updating means that neighboring nodes in the output space are pulled into the same input space area as the winning node, thus producing an ordered mapping from the input space to the output nodes. The result is that inputs from similar parts of the input space should be classified by nodes from similar parts of the output lattice.

The SOM provides a classification that, because of the structure imposed on the output nodes, is easier to interpret meaningfully than the CLNN. The movement of the neighborhood nodes lessens the effect on the SOM of some of the problems met by the CLNNs, specifically dead units.

2.1 Soft Competition

Standard competitive learning neural nets use gradient descent to reduce the SSE, as defined in equation (3), aiming to find the global minimum, or a good local minimum that is close to the global minimum. Unfortunately gradient descent methods are highly dependent upon the initial state of the network and the minimum found is frequently a poor local minimum, which often manifests itself as dead units in a CLNN. Several researchers have proposed the addition of stochasticity, often in conjunction with a simulated annealing technique, as a means of overcoming this difficulty for competitive networks. One of the best of these clusterers is the Neural Gas model of Martinez [Martinez et al., 1993] which uses soft competition to help avoid local minima.

Soft competition involves updating all the nodes in the network to some degree as opposed to just the winning node. This movement of all the nodes, although computationally expensive, helps to improve the performance in two ways. Firstly, the movement of all nodes helps to prevent dead units, this effect is similar to that provided by the use of a conscience or leaky learning mechanism [Hertz et al., 1991], and this alone will improve performance to some extent. The second factor that improves performance is that the change in the value of the SSE is not just in the direction of the steepest gradient but may cause a jump to a new value somewhere else in the error space. Initially the movement around the error space can be quite high but is reduced with time, by using the analogy of a decreasing temperature coefficient, similar to the simulated annealing approach. This means that at high temperatures the network can jump out of poor local minima, and then gradually settle on a solution.

The Neural Gas network uses soft competition and a temperature coefficient to effect the softness of the competition over time. The network minimises the following cost function:

$$E = \sum_{\mathbf{x}} \sum_{i=1}^{N} e^{\beta(k_i(\mathbf{x},\mathbf{w}_i))} \left|\mathbf{x} - \mathbf{w}_i\right|^2 \qquad (4)$$

where:

β = 1/T, T is the temperature,

$k_i(\mathbf{x},\mathbf{w}_i)$ is a function which represents the ranking of each weight vector $\mathbf{w_i}$ with each input vector $\mathbf{x}$. If $\mathbf{w_i}$ is closest to input $\mathbf{x}$ then k = 0, for the second closest k = 1 and so on.

At high temperatures many of the nodes in the network add to the cost (error) of the network. As the temperature is reduced only the nodes close to the input contribute significantly to the error, and at very low temperatures effectively just the winning node produces an error. Thus the cost function is reduced to the SSE cost function in (3) for very low temperature values.

The corresponding update rule, that is applied to all nodes and which reduces the cost function in (4), is:

$$\mathbf{w_i}(n) = \mathbf{w_i}(n-1) + \alpha\, \mathrm{e}^{-\beta(k_i(\mathbf{x},\mathbf{w}))}(\mathbf{x} - \mathbf{w_i}(n-1)) \qquad (5)$$

where:

α is the global learning rate of the network in the range 0 to 1 and

n is the time step.

The Neural Gas network is computationally expensive. The nodes all have to be ranked and updated for each input vector. At low temperatures the size of the update for the majority of the nodes is minuscule and will have no real influence over the network although the computational requirements remain unchanged. In a previous study [Butchart et al., 1995] we investigated three networks that used soft competition, the Neural Gas network, the Generalised Learning Vector Quantisation (GLVQ) [Pal et al., 1993] and the Deterministic Soft Competition Network (DSCN) [Yair et al., 1992]. We concluded that of the three networks the Neural Gas network was by far the superior in all round performance. It also performed better than standard competitive networks. We showed that the Neural Gas network provided a reliably good SSE over many different data types, and was resilient to initial starting positions and variations in parameter settings. This is, however, gained at the expense of an increased computational load and slower convergence rate, though it was able to produce fair results even when given a limited time in which to converge. The dimensionality and scale of the input space does not appear to unduly influence the performance of the network.

Soft competition has partially solved the tendency of standard competitive nets to fall into poor local minima, but other problems associated with competitive nets still exist. Most specifically soft competition networks still require the user to decide in advance how many output nodes are to be used. Soft competition networks also do not place any structure on the output nodes which make the results produced by the network difficult to interpret meaningfully. The dynamic neural tree networks discussed next address these two problems.

2.2 Dynamic Neural Tree Networks

Dynamic Neural Tree Networks (DNTNs) autonomously decide upon the number and structure of output nodes required and provide a scaleable hierarchical classification of the data set. Li et al developed a version of the dynamic neural tree network and Racz and Klotz independently developed a very similar tree growing network [Li et al., 1992, Racz and Klotz 1991].

DNTNs are single layer feed forward networks where a hierarchical tree structure is superimposed on the output layer nodes. The output nodes are created dynamically and each is fully connected to the input layer. The output nodes are organised into a tree structure, in which all nodes, except the root node have a single parent. The children of the current node (the previous winning node) are the only ones allowed to compete to classify the input. The winner is updated as in (2) and its children then recursively compete to classify the input.

The algorithm for input classification (assuming no growth is required) is as follows:

1. The root node is designated the current node;
2. The children of the current node compete using equation (1) to classify the input, the winning child is designated the next current node;
3. The current node is updated to move it closer to the input using equation (2);
4. Steps 2 to 3 are repeated until the winning node has no children;
5. Steps 1 to 4 are repeated for each input.

The final structure of the network is not restricted to a binary tree. An example of a possible tree structure is shown in Figure 1, each node is connected to the input layer, and there are connections between parent nodes and their subtrees.

DNTNs have associated with them two parameters, known as the tolerance and threshold, which have a strong influence over the structure of the tree created.

The tolerance parameter defines the radius of the classifying hypersphere of every node at each level. The size of the tolerance parameter decreases as you travel down the tree away from the root node; so that nodes towards the bottom of the tree provide a finer grained classification than those at the top of the tree. An input that is outside of the tolerance for the nearest node causes a new sibling to be created. This automatic generation of a sibling for each input that is out of tolerance leads to run-away sibling growth for data that contains many outliers.

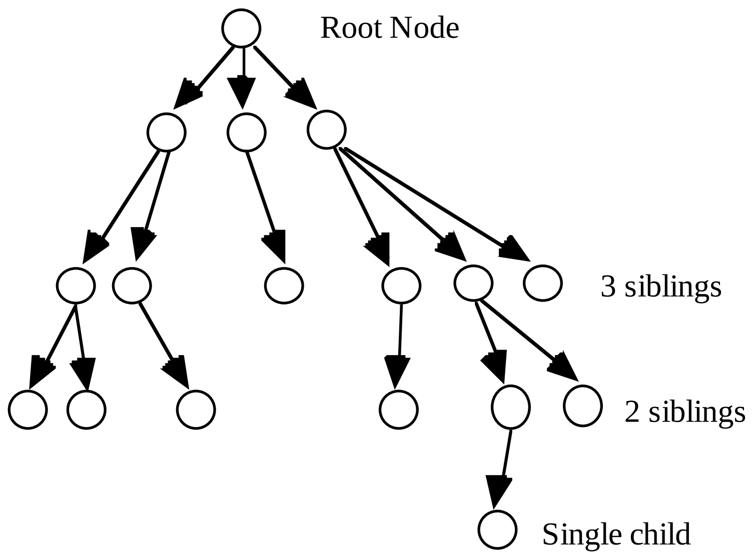


Figure1. Typical tree with child nodes and groups of sibling nodes shown

The threshold parameter determines when a new child node is created. The threshold values reduce as you move away from the root node to make new child growth less likely at lower levels in the tree. The Li network stores information on the cumulative error a node has produced - if this exceeds the threshold value a new child is created. At each level in the tree the error is compared to an absolute value so this technique is referred to as an *absolute error* mechanism. This error is recalculated each time the node is active and its value depends on whether the node is frequently active within a window of time. Thus nodes that are frequently active will tend to have a greater error than those that are only active occasionally. The Racz and Klotz network compares the activity of a node (how often it has classified any input) to the activity of its parent node. If the ratio between the child node and the parent node's activity values exceed the threshold then a new child is created. At each level in the tree the activity of a node is compared to the value for its parent so this technique is referred to as a *relative activity* mechanism.

The tree structure is built dynamically in response to the structure inherent in the data set. Initially the neural tree consists of a root node. When the first input is presented to the network a child of the root node is created to classify the input. This node's weight vector being set equal to that of the first input. As the rest of the inputs are presented to the network the following growth algorithm is performed for each input vector:

1. Initially the children of the root node compete to classify the input;
2. If the winning node is within tolerance of the input its weight vector is updated using equation (2).
    a) If the winner has child nodes then the children compete to classify the input and the algorithm recurses.
    b) Else if the threshold of the winner is exceeded then a new child node is created;
3. Else the winning node is not within tolerance of the input and a new sibling is created to classify this input.

In the course of the development of the tree structure some nodes are likely to be created in positions that prevent them from winning the competition to classify nodes. These nodes are dead and there is no mechanism within the basic DNTN paradigm to force them into the competition. These dead nodes can be removed from the tree by a pruning procedure that deletes nodes with insufficient activity.

In our initial study of these networks [Butchart et al., 1996a] the Racz and Klotz network produced some very good performances and some exceptionally poor ones. The problems with the poor results were caused by unsuitable tolerance and threshold values. The poor parameter values cause an elongated tree structure to be produced, and emphasized the problem this network has of tending to grow a child from the first node produced in a set of siblings when the growth is not necessarily justified. The resultant tree structures are totally imbalanced, producing poor SSEs and failing to represent the nature of the data set. If the parameter values are set to a more suitable level the network can produce a much better performance.

In the same study [Butchart et al., 1996a] the Li network generally produced good solutions when given suitable parameter values. When the network was given suitable parameter values a well balanced tree was produced. The Li network is generally more stable than the Racz and Klotz network in the production of new nodes, but suffers from the same problem of generating new siblings for each noisy input .

Both the Li network and the Racz and Klotz network demonstrated their potential to produce fast scaleable hierarchical solutions if they were given suitable parameter values. The tests also highlighted the sensitivity the networks have to their parameter values and the unstable, unrepresentative growth that can occur if the values are incorrectly set. There are no heuristics provided by the authors of the two networks to guide the search for suitable parameter values. This lack of heuristics meant that the search was performed by guessing a suitable starting point and then assessing the results to guide the search. If the nature of the data set is known then the search may be quite quick and a good solution found at the 3rd or 4th attempt. As already mentioned, both networks suffered from automatic sibling growth when presented with data with noisy inputs (outliers).

2.3 Hierarchical ART Network

Another issue relating to adequate tree-structure growth is the ability of the higher level (non-classifying) nodes to provide meaningful hierarchical information about the data. Here we intend to compare the super-ordinate clustering ability of the SCENT network with that produced by another hierarchical clusterer - the hierarchical version of the ART network, HART [Bartfai 1995].

The HART network consists of layers of ART networks [Carpenter and Grossberg 1987] where each successive layer learns to cluster the category prototypes developed at the layer below it. ART itself consists of layers of neurons and, apart from the first ART network, the input layer of each subsequent ART network is the comparison layer of the previous one. Consequently if the input vector for the lowest ART network is $\mathbf{x}$ then the input for the next ART layer is $\mathbf{w}_\mathbf{I} \cap \mathbf{x}$ where I is the node found in the category layer of the first ART network that matches the input pattern.

The algorithm for training the HART network is:

1. Initialise each ART network;
2. Present the input vector $\mathbf{x}$ to the first input layer (the lowest ART module)
3. The ART module categorises the input by finding a node I in its category layer to match the input;
4. The weights for the winning node I are updated according to the usual ART update rule:

$$\mathbf{w}_\mathbf{I}(n) = \eta(\mathbf{w}_\mathbf{I}(n-1) \cap \mathbf{x}) + (1-\eta)\mathbf{w}_\mathbf{I}(n-1) \qquad (6)$$

where:

η is the learning rate ($0 < \eta \leq 1$) and n is the time step;

5. Steps 3 and 4 are repeated for each ART module (this could be done in parallel with dealing with the first ART module but was implemented as a sequential search for simplicity);
6. Steps 2 to 5 are repeated for each input.

As a result of the above learning procedure higher layers develop fewer and more general categories of the input patterns while the lower layers learn more specific categories. Hence the outputs of the category layer of each ART module shows which class the current input belongs to at each level in the class hierarchy defined by the modules.

3 The SCENT Model

SCENT [Butchart et. al., 1996b] shares some of the characteristics of other dynamic neural tree models (DNTNs), described earlier. However the criteria for growth, the addition of pruning and the stochastic nature of the model are different. In addition SCENT has no external parameters to be set for runs over different types of data and the internal behaviour of the model is stable with respect to the values of its internal constants. This has a significant benefit in overcoming some of the problems associated with the other DNTNs described earlier.

The main features of SCENT are:

- Nodes are selected for growth on the basis of relative activity
- The nature of the growth (sibling or child) is decided after selection for growth

- New growth may be pruned quickly if unsuccessful in reducing the classificatory error of its cluster
- Nodes that have prolonged low activation are pruned
- All pruning is undertaken stochastically

SCENT is a development of an earlier model CENT which did not have a stochastic element. The addition of mediated non-determinism has added to the stability of the resulting clusterings, as will be further discussed in section 4.1.

3.1 Classification and Learning

This is performed in SCENT exactly as in the other DNTNs as described in section 2.2. We use a learning rate of 0.05. We have not found it necessary to vary this setting or any details of the learning schedule in any of the experiments reported here. The input vectors are presented to the tree in epochs, but the order of presentation within an epoch is random and different for each epoch. In general the number of epochs needed for convergence was found to be small (of the order 15-35 epochs).

3.2 Growth and Pruning

3.2.1 Initialisation of the Root Node

The tree begins as a single root node. An initial pass through the data is used to establish the position of the root roughly at the mean of the data and simultaneously the initial tolerance of the root is set so that roughly two-thirds of the data lies within the hypersphere with radius equal to the root tolerance. At the end of this initial pass the root node produces two children.

3.2.2 Selection For Growth

A node can only be selected for growth if it is currently a winner and a leaf. As well as storing its threshold a node also maintains an activity count, roughly speaking, a count of the number of times the node has won. However in order for the performance of the network to be relatively invariant with respect to the order of presentation of the input vectors the activity is linearly decayed over time, as originally proposed by Li [Li et al., 1993]. The decay rate is chosen so that if a node has not been active for one third of an epoch then its activity will have decayed to zero.

A node is chosen for growth if its *relative activity*, the ratio of its own activity to that of its parent, exceeds a threshold. Rather than use a fixed threshold as in the DNTN of Racz and Klotz [Racz and Klotz, 1991] we require that a node has relative activity greater than the average of its siblings:

$$\text{act(node)} * \#\text{siblings} > \text{act(parent)} \qquad (7)$$

where
act(node) is the current activity of the node.

In fact we relax this criterion and use:

$$\text{act(node)} * (\#\text{siblings} + 4) > \text{act(parent)} \qquad (8)$$

This has the effect of encouraging much growth; the addition of aggressive pruning keeps the tree healthy and concise.

3.2.3 Nature of Growth

Having decided to grow, a leaf node must decide whether it should spawn children or a sibling, and to do this it makes use of its tolerance value, the radius of its classificatory hypersphere. If the majority of vectors classified by the node are within tolerance then it is assumed that the node is classifying a spatially compact group of vectors and growth is downwards, two children are produced. On the other hand if most vectors are outside tolerance the local data is spatially separate and a sibling is produced.

3.2.4 Growth

The new node is initialised with a weight vector that is a noisy copy of the parent, or sibling if it results from sideways growth. In order to provide a classification of finer granularity in lower plys of the tree the tolerance of a new node is a reduction of its parent’s. The actual reduction is contingent upon the success of the parent in classifying vectors within tolerance. If most activity is within tolerance then the reduction approaches 50%. It is also worth noting that nodes deeper in the tree will also have lower activities simply because fewer input vectors will reach them.

When a node first produces children its activity count is zeroed so that any future relative activity calculations use relevant data.

3.2.5 Stochastic Pruning

The SCENT model works under the premise that growth should be encouraged, and as a consequence ineffective growth must be pruned back. Two forms of pruning are used. Short term growth rejection takes place if a new node does not reduce the classificatory error of its parent sufficiently; this decision is taken one epoch after the node is created.

A leaf node that has established itself in the tree may still be removed if its long term performance is inadequate. This is measured by comparing the activity of the node against a threshold that decreases in lower levels of the tree; this operation is only performed at epoch boundaries. Consequently any leaf node with low activity can potentially be pruned several epochs after creation.

It is in the pruning process that stochasticity is used. In early growth it is useful for the network to create much tentative new growth, and also for longer term pruning to be more common. To this end *growth rejection*, is initially made less likely - it is a probabilistic process, inversely proportional to a temperature value that is initially high and decreases exponentially over time. Similarly long term pruning is made more likely early in a node’s life - it is stochastic, but directly proportional to temperature. The addition of this mechanism leads to a more reliable performance by the model when compared to its non-stochastic predecessor.

3.3 The Algorithm

A summary of the SCENT learning and pruning algorithm is:

```
1. Read in vectors and initialise root node
2. Do growth phase
While (more epochs required & sufficient growth occurring)
    Shuffle vectors
    For (each input vector)
        For (each level of the tree)
            Find winner using equation (1) and adjust its position using equation (2).
            Update activity and error measures
            If (growth allowed using equation (8))
                If  (most classified vectors are not within tolerance)
                    grow sibling
                Elseif (most classified vectors are within tolerance)
                    grow child
            End if
        End for
        Accumulate error
    End for
    Stochastic pruning and deletion of unwanted nodes
End while
3. Adjust tree phase
While (error change sufficient)
    Shuffle vectors
    For (each input vector)
        For (each level of the tree)
            Find winner and adjust its position as before. Update error measure
        End for
        Accumulate error
    End for
End while
4. Do final node deletion
```

4 Results

The performance of CENT and SCENT as hierarchical clusterers, particularly their ability to discover hierarchical structure, has been evaluated. Both models were evaluated comparatively (section 4.1) allowing an empirical judgment of the ability of the two networks to be made over a range of data sets. Results of CENT and SCENT were compared with each other and with the three other neural models described in this paper: Neural Gas [Martinez et al., 1993] , Li's Neural Tree [Li et al., 1993] and Racz and Klotz's dynamic learning model [Racz and Klotz 1991]. SCENT was also evaluated structurally (section 4.2) using two further data sets to determine the representational aspects of SCENT's clustering capabilities. A brief comparison of SCENT with HART [Bartfai, 1995] was also made for the second of these data sets.

4.1 Comparative Evaluation

Each model was run over each data set (see table 1) for 15 complete runs. Experimentation was initially performed to find a satisfactory set of parameters for all the networks except CENT and SCENT, as these models do not require parameter setting. In particular the number of nodes in the Neural Gas model was set to be roughly that number of the leaf nodes that SCENT produced, so that the final SSEs are directly comparable. The results produced are averages over these 15 runs. The averaged results are summarised in table 2.

4.1.1 Data Sets

The data sets have been carefully chosen to test the networks over a wide range of different performance areas, including the sensitivity of the networks to alterations in the dimensionality of the data. Many of the data sets are in 2 dimensions to enable a visual interpretation of performance, however there are also two higher dimensional data sets. Data has been produced that can be viewed hierarchically to test the performance of the hierarchical networks. The nature of the majority of these data sets is known as they have been generated especially to test many aspects of the clusterer's performance in this experiment. Data sets 5 and 6 are illustrated in Figures 2 and 3. The IRIS data set is included to provide a benchmark performance.

Set 1	2-D single source Gaussian cluster, zero mean and unit variance.	Simple cluster, base line test.
Set 2	20-D single source Gaussian cluster, zero mean and unit variance.	Simple cluster, dimensionality test.
Set 3	2-D Uniform distribution within a square	No cluster test.
Set 4	2-D data, 6 even clusters in 4 groups.	Balanced hierarchy test.
Set 5	2-D data, 10 clusters with varying density and size.	Unbalanced hierarchy test.
Set 6	2-D data. Varied Clusters with hierarchical structure	Informal hierarchy test.
Set 7	Anderson's IRIS data [Everit, 1993]	General cluster performance benchmark.

Table 1. Comparative Study: Data Sets Description.

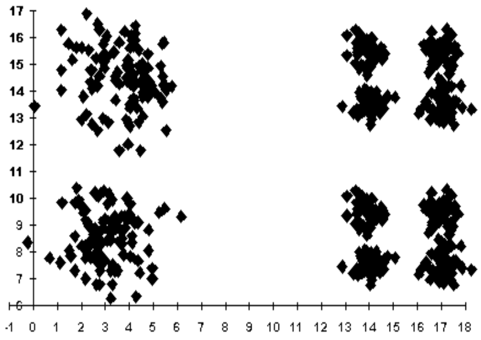


Figure 2. Data set 5: 10 clusters in 4 large groups forming an uneven hierarchy.

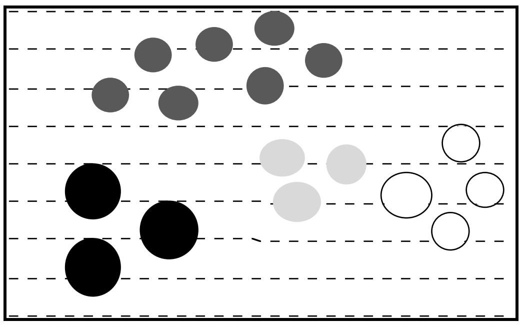


Figure 3. Data Set 6 : The positions of the 17 Gaussian clusters present in the two dimensional test data set. The shading of the cluster positions indicate a potential hierarchical grouping of the clusters.

4.1.2 Comparative Results

It is very difficult to judge the performance of a hierarchical cluster, but relatively straightforward to measure the quality of a flat classification. The aim of a clusterer is to produce low Sum Squared Error (SSE) but in a dynamic network this is complicated by the ability of the net to use an unbounded number of nodes. A more reasonable comparison for flat and hierarchical clusterers is, therefore, between the product of SSE and number of nodes. The results are summarised in table 2, in which the column marked “*” is the product of SSE and the number of nodes in each network.

	SCENT			CENT		
Set	**SSE**	**Nodes**	*	**SSE**	**Nodes**	*
1	20	28	560	22	28	616
2	678	27	18306	677	28	18956
3	21	29	609	19	38	722
4	248	25	6200	250	29	7250
5	330	27	8910	559	19	10621
6	129	28	3612	123	30	3690
7	2	36	72	2	24	48
Average	204	28.6	5467	236	28	6608

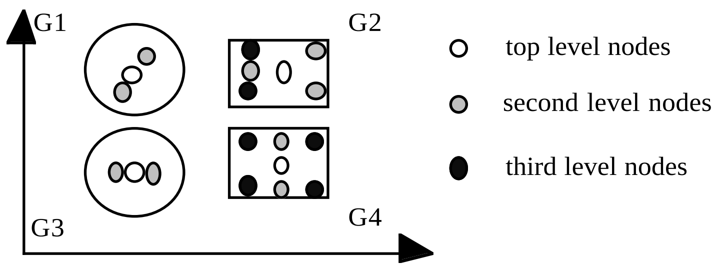
Table 2a. Comparison of CENT and SCENT, with results averaged over 15 runs. The column labeled “*” is the product of the SSE and number of nodes, the preceding two columns.

	NGas			Li			RK		
Set	**SSE**	**Nodes**	*	**SSE**	**Nodes**	*	**SSE**	**Nodes**	*
1	15	16	240	22	26	572	24	57	1368
2	679	16	10864	766	110	84260	346	685	237010
3	11	25	275	11	43	473	48	20	960
4	125	32	4000	194	55	10670	602	42	25284
5	272	16	4352	298	49	14602	455	45	20475
6	132	16	2112	125	40	5000	79	56	4424
7	1	16	16	6	8	48	3	32	96
Average	176	19.6	3123	203	47.3	16518	222	134	41374

Table 2b. Comparison of three other Neural Models, with results averaged over 15 runs. RK refers to the Racz and Klotz neural tree.

The results of a typical run of SCENT (in this instance over data set 5, figure 2) can be seen in Figure 4. The resultant tree structures produced by SCENT varied across runs but were generally consistent. It can be seen that SCENT has produced four top level nodes, positioned at the centroid of the four large data groups. For the two unstructured clusters on the left, G1 and G3, two further nodes are produced at level 2. G2 and G4, on the right , have more structure and this is reflected in the subtrees produced by SCENT, although the result is not an exact match to the visible structuring.

(a)



(b)

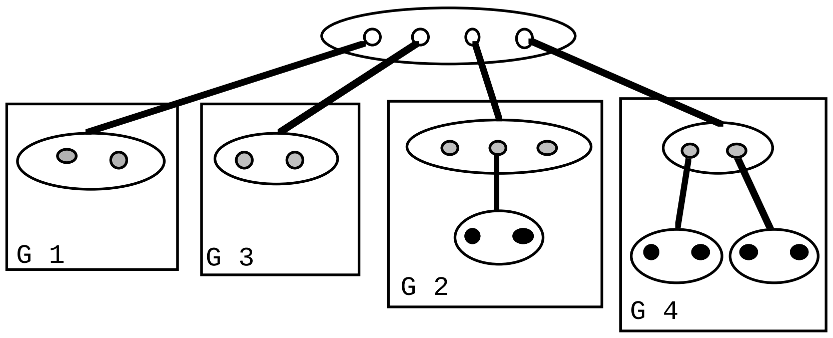


Figure 4. (a) The position of nodes in a typical run of SCENT, (b) the organisation of the nodes in the resulting tree. Results from data set 5, pictured in Figure 2.

4.1.2.1 Comparison of CENT and SCENT Models

Comparing the results of CENT and SCENT (see table 2a) over the 7 data sets it can be seen that the addition of stochasticity to the growth and deletion procedure of CENT did not have a major effect on the average performance of the network. The best and average results from a group of runs over a data set were often similar between the standard CENT model and the model with stochasticity added. There was however often an improvement in the worst case result, with the worst case result often being significantly better for Stochastic CENT (SCENT) than for the standard CENT. This was seen as a positive improvement as it increases confidence in a network if very poor performances can be eliminated.

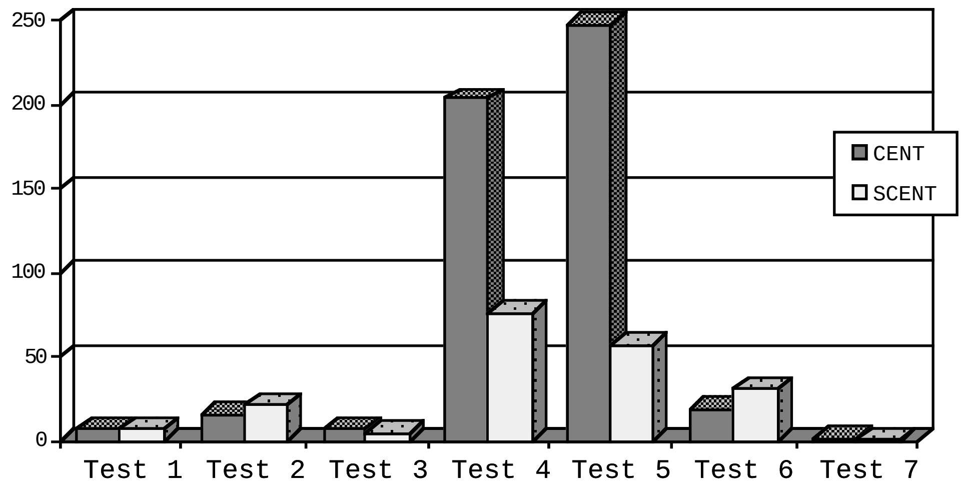


Figure 5. Standard deviation in SSE results for the SCENT and CENT models over tests 1- 7.

The problem involving worst case results encountered with the CENT network leading to a poor degree of repeatability of results under different run time randomisations is illustrated in Figure 5 where CENT and SCENT can be compared for each set of test data. This lack of repeatability was particularly apparent over the hierarchical test sets (tests 4 and 5). Over these test sets some of the CENT results were close to the optimal whereas others were very poor. It was hoped that the stochastic additions in the SCENT model would help to avoid the very poor performances whilst maintaining the quality attained by the very good performances. Figure 5 shows the standard deviation of the SSE results for the 15 runs performed for both the SCENT and CENT models over the 7 test sets. It can be seen from Figure 5 that although the SCENT has caused a slight increase in the standard deviation over two of the test sets it has significantly reduced the range of performances over the hierarchical test sets.

An unexpected side effect of the alterations in the node growth and deletion procedures was the increase in convergence rates for the SCENT model in comparison to the CENT model, shown in Figure 6. This increase in convergence rates more than compensates for any extra computation required in the SCENT model.

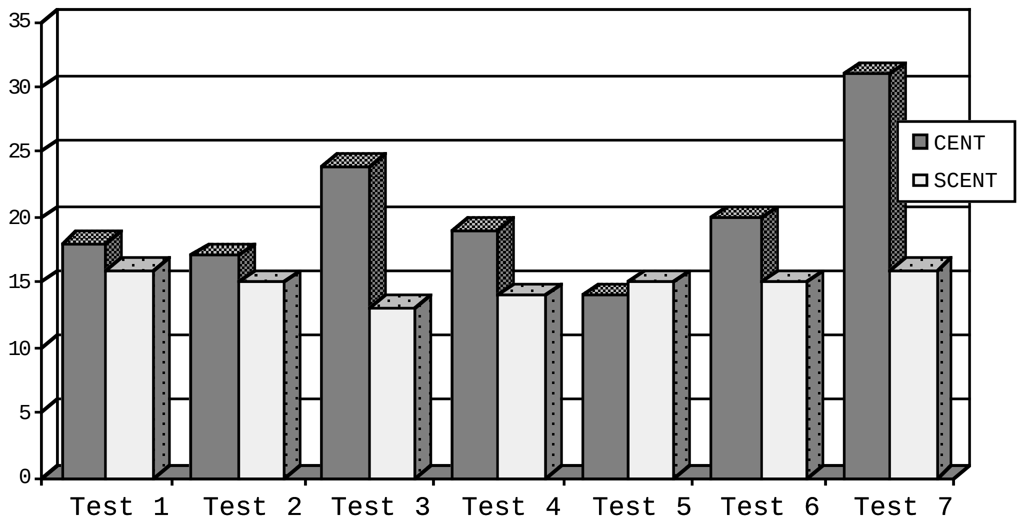


Figure 6. The average pass on which convergence was reached for the SCENT and CENT models over tests 1- 7.

4.1.2.2 Comparison to other network performances.

The SCENT model has been compared with two other dynamic neural trees, the dynamic competitive learning technique of Racz and Klotz [Racz and Klotz, 1991], and the Neural Tree of Li [Li et al., 1992] and also the flat clusterer Neural Gas [Martines et al., 1993], see Table 2b. Apart from technical differences between the models SCENT has one fundamental difference: it requires no initial parameter setting.

The presentation of the results is to some extent unfair to the tree based networks as many nodes are not used as classifiers, and in this light the performance of SCENT is exceptionally strong. Its average SSE multiplied by #nodes is roughly double that of Neural Gas, which implies that the leaf nodes are almost certainly out-performing Neural Gas. The performance gap to the other two dynamic trees is substantial.

When attempting to judge the quality of the hierarchies produced by the tree based networks the criteria to look at are more subjective. However for the data which is designed to be hierarchical we would like the induced structure to reflect that of the data, so that, for example, the top level of the tree should contain representatives for each of the major groups in the data and subsequent layers should reflect subgroupings. Some behaviour is not desirable, specifically a tendency to produce long thin branches which do not add to the semantic content of the tree. It was found that both Li and Racz and Klotz had a tendency to do this which could only be avoided by very careful selection of parameters, although even this was not always possible. The very high average node count for Racz and Klotz is explained by this phenomena. A typical example of the classification tree produced by SCENT for data set 5 is shown in Figure 4, together with the position of the classifying nodes. It can be seen that the network has produced a near optimal classification and a reasonable tree structure.

(S)CENT in comparison to Neural Gas

Over the test sets consisting of regular 2 dimensional data, especially test 1 and test 3 the Neural Gas performance is significantly better than the (S)CENT models. This is because the Neural Gas network is particularly well suited to this type of data, whereas the (S)CENT models find this type of data set amongst the most difficult to classify.

Over data sets with high dimensionality (test 2) or with more groupings amongst the data (tests 4, 5 and 6) the performance gap between the networks lessens.

The Neural Gas and (S)CENT models are very different types of networks offering different functionality. Neural Gas provides good error reduction classifications, but as a static flat network it does not provide easy interpretation or a hierarchical analysis of the data. The (S)CENT models, as dynamic tree structured networks, provide a hierarchical and more interpretable analysis of the data being classified. The use of a dynamic tree structure causes difficulties for the models in classifying certain types of data. The fact that the (S)CENT models have produced performances that are at times close and always within a fair range of the Neural Gas network can be seen as a good result for the (S)CENT models given the differences between the network types, and the quality of NGas as a flat clusterer [Butchart et al., 1995].

The Neural Gas network is more computationally expensive than the SCENT models. The larger the data set the more noticeable the difference in the time required to produce the classification. Further experimentation showed that with visual data (such as that described in 4.2.1.1) the Neural Gas network would require over-night run times whereas the SCENT model would classify the data in 30 to 40 minutes. This vast run time difference gives an indication of the difference in computational expense.

(S)CENT in comparison to Li

The Li network has a comparative advantage over the (S)CENT models, for all tests runs, in that time has been spent optimally presetting the parameters of the Li network. This time was spent in order to help the Li network produce performances as close to the known optimum as possible. The (S)CENT models do not have parameters to be set by the user and make their own decisions based on the data set. Even given this advantage the Li network only produced a better average performance on the uniform data set of test 3. The performances of the networks were similar over tests 1, 6 and 7 the (S)CENT performance was better over tests 2, 4 and 5.

(S)CENT in comparison to Racz and Klotz

As with the Li network the Racz and Klotz network had the advantage of having the parameter values altered to enable the network to produce a result close to the known optimal result. The Racz and Klotz network did not however produce good performances. The (S)CENT models bettered the Racz and Klotz performance over tests 1, 2, 3 and 5. Over tests 6 and 7, data sets which the Racz and Klotz network is more suited to, the performances were comparable but the (S)CENT models were often slightly better.

4.2 Representational Evaluation

The tree structure of the SCENT model was evaluated using visual data (4.2.1.1) and semantic data (4.2.1.2) both containing no explicit hierarchical structure. Each data set was presented 4 times to the SCENT model and the resultant tree structures were analysed; averages over 4 runs are also presented, see table 5.

4.2.1 Data Sets

4.2.1.1 The Picture Data

This data set consists of 153 vectors [Gale 1997]. Each vector has 2500 grey scale elements in the range 0-255 representing a 50*50 grid of pixels. The vectors have been formed by scanning a set of pictures with care being taken to ensure evenly sized and centralised images. There is no semantic information included in the input vectors, which consist purely of the grey scale images. There are 17 categories of image, such as snakes, birds, fish, clocks, tables and guitars. There are 3 varieties of each category with the exception of pianos (with 2) and cabbages/lettuces (with 4) giving 51 different images, each image occurs in 3 different contrast variants. The high contrast set is shown in figure 7.

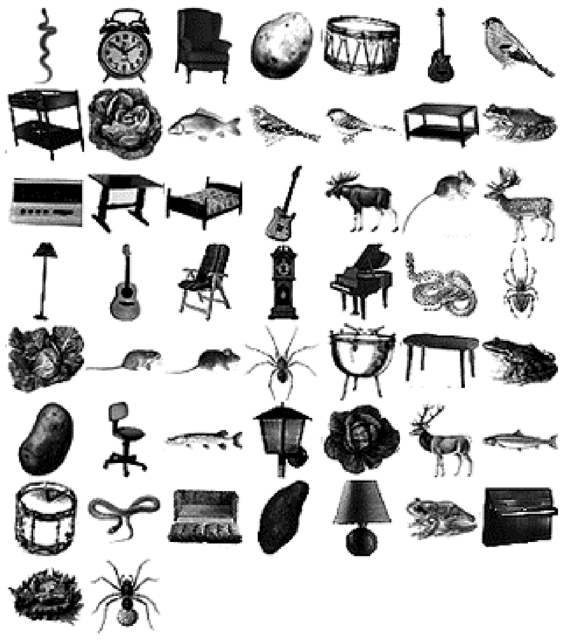


Figure 7. The 51 high contrast pictures from the 153 picture data set.

In the grey scale images white is represented as 255 and black is 0. As can be seen from the set of images there is some variation in general tone (e.g., fish, birds, mice etc. are lighter and armchair, cabbages, upright piano etc. are generally darker). The lighter images are in general smaller (10% (250 pixels) to 40% (1000 pixels) of the image). The darker images are generally larger (20% to 80% of the image).

From this it can be seen that, since white is represented as 255, the difference in vector component for an image with a non-white pixel as against a white pixel is larger than the difference in vector component for two contrast variants of the same image, considerably larger in the case of a dark contrast darker image. Hence two different images will generally have a larger distance from each other than will two different contrast variants of the same image.

4.2.1.2 The Zoo Data

The Zoo machine learning database [Murphy & Aha, 1992] describes 101 instances of animals in terms of 18 attributes: a naming label “animal name”; 15 binary attributes such as: aquatic or venomous; a field giving the number of legs, an integer between 0 and 8; and a type label. The type label indicates a biological taxonomic group dividing the 101 instances into 7 categories of animal, each uniquely labeled by an integer. Six of the categories clearly define large taxonomic groups, such as: mammals, birds and insects. The seventh categorises other animals: clams, crabs, crayfish, lobsters, starfish, octopus, scorpion, seawasp, slug and worm. An example input vector for the platypus is presented in table 3, below.

Data was presented to SCENT in two formats, typed and untyped. The typed data set contains 17 of the full 18 attributes, it excludes only the animal name from the database. The untyped data set also does not contain the animal name label but more significantly the type attribute was removed and the number of legs attribute normalised.

Feature	Label	Value
2	Hair	1
3	Feathers	0
4	Eggs	1
5	Milk	1
6	Airbourne	0
7	Aquatic	1
8	Predator	1
9	Toothed	0
10	Backbone	1
11	Breathes	1
12	Venomous	0
13	Fins	0
14	Legs	4
15	Tail	1
16	Domestic	0
17	Catsize	1

Table 3. The input vector for the platypus instance in Zoo data set.

4.2.2 Representational Results

4.2.2.1 Picture Data Results

Figure 8 shows a typical result formed by using SCENT on the 153 vector data set. Each final and non final node is represented in the tree by its classifying weight vector converted back into a 50*50 pixel image. It generally collects the 3 contrast variants of the same image together despite the fact that they are presented in a random order each epoch. The reason for this can be seen by considering the metric used by SCENT to determine the nearest vector for classification purposes. This metric is the Euclidean distance formula:

$$d = \sqrt{\sum_{i=0}^{2499} (x_i - w_i)^2} \qquad (9)$$

where d is distance, x_i is the ith component of the input vector and w_i is the ith component of the weight vector.

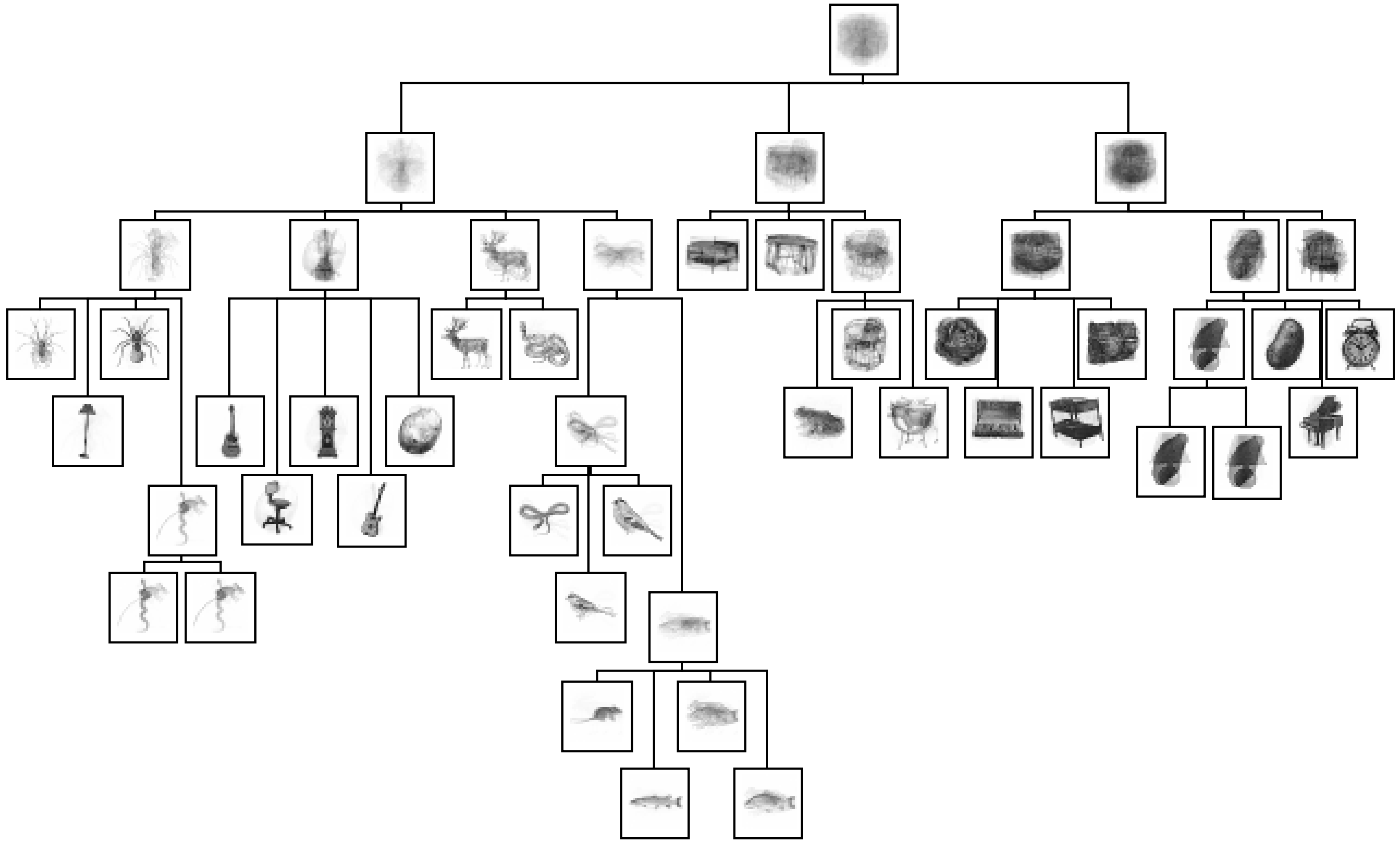


Figure 8: A tree produced by SCENT, for the picture data showing the weight vectors of the nodes displayed in the same format as the input vectors

In the lighter images the darkest contrast variants average pixel values around 125 and the lightest contrast variants average around 175. In the darker images the darkest contrast variants average pixel values around 50 and the lightest contrast variants average around 90.

The pixel values (255 white, 0 black) explain the ability of the model to separate lighter and darker images, since for each of the dark images the difference between a white pixel with no image and a pixel with an image is considerable (up to 80% of the component's possible contribution) whereas a lighter image is generally smaller and each pixel is less different from white.

Figure 8 also illustrates one of the benefits of using this set of data. The visual character of the data makes it possible to see the nature of the analysis provided by the network. It can be seen that the weight vectors move from rough generalisations at the top of the tree structure to more detailed pictures at the lower levels. In general the tree splits the lighter and darker images into separate sub-trees. The left hand sub-tree has classified the lighter and smaller images and the other two sub-trees classifying darker and larger images with the right hand sub-tree classifying the darkest images. The next level nodes classify according to the general orientation of the image. For example in the left hand sub-tree the first and second classifying vertical images; the second deals with the darker variants; the third classifies the more diagonal elements; and the fourth classifies the horizontal images.

In two cases two leaf nodes appear to be the same as their parent node. This is partly because the parent nodes are classifying two images and the full sized images show more of a difference and also that the child nodes are obviously newly created and have not yet had time to differentiate properly.

4.2.2.2 Zoo Data Results

Figure 9 shows a typical tree produced using SCENT with 101 untyped vectors of the Zoo data set. Each of the data vectors is clearly categorised at both super and sub-ordinate levels. The tree structure produced using each data set was labeled in two ways. Firstly, the data vectors represented by each leaf node of the tree were labeled with the associated animal name tag. Secondly, the semantic type label of each instance was examined and the leaf node and super-ordinate clusters were classified according to each of the 7 labeled groups. Each super-ordinate class is further classified at each subsequent level of the tree. All instances of mammals are grouped together, in the leftmost clusters of the tree, and are subdivided into large-predatory, large-non-predatory, small, aquatic-legged, aquatic-finned. The middle cluster categorises fish and birds. The birds are classified in two subtrees, as predatory or non-predatory. The right hand cluster represents a variety of vectors from the less well represented classes of animal. It can therefore be seen that the emergent clusters produced by SCENT often correspond with natural taxonomic groups.

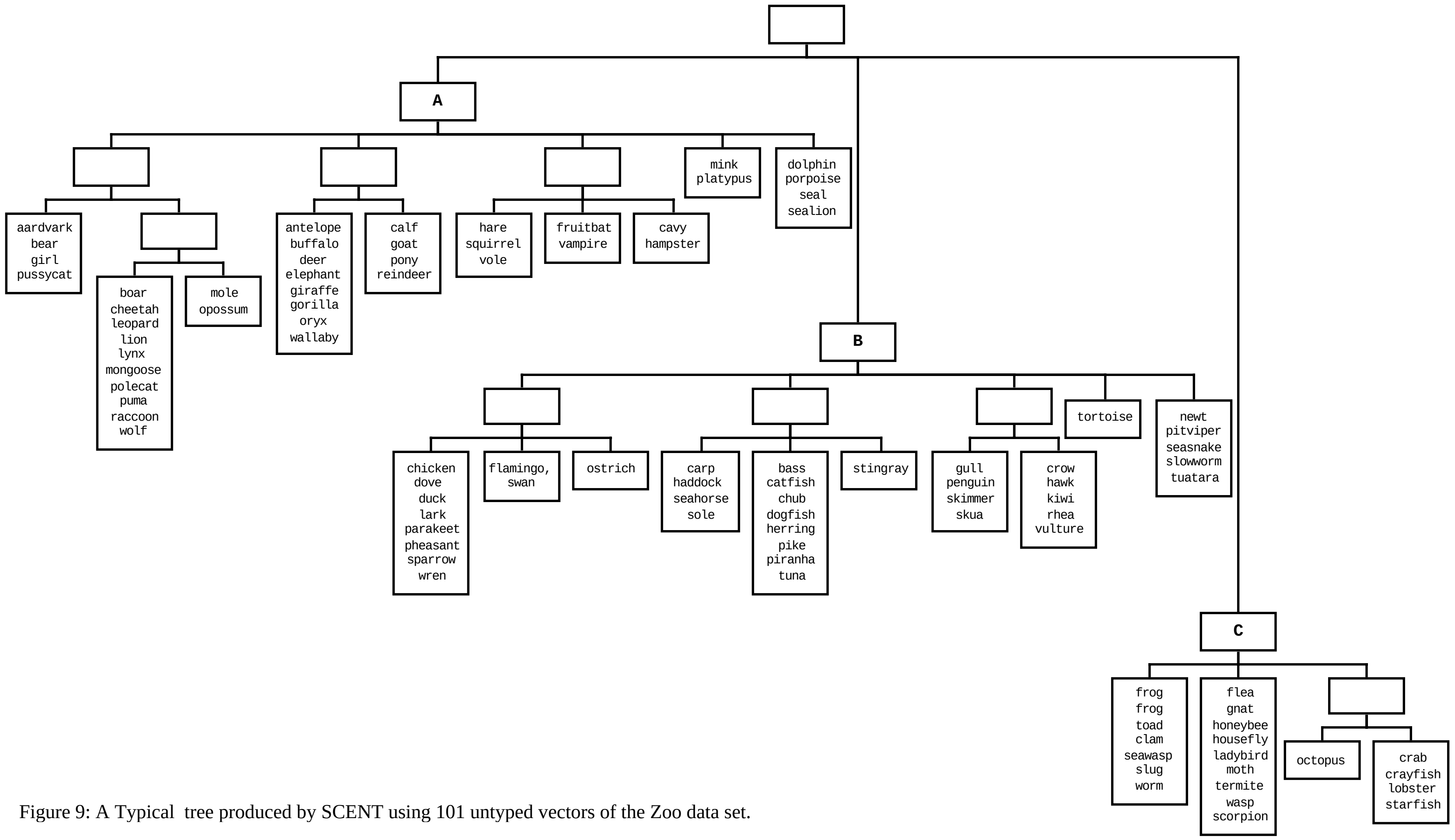


Figure 9: A Typical tree produced by SCENT using 101 untyped vectors of the Zoo data set.

The SCENT model produces similar tree structures to that shown in figure 9 for both the typed and untyped data sets. It is interesting to note that the absence of the type label, in the untyped data set, has little impact on the structure of the tree or its hierarchical classification. The structures of the trees produced by SCENT are highly similar when clustering both typed and untyped data and the classification hierarchy has the same super-ordinate clusters as those identified in the type label. Typical misclassifications seen in SCENT, such as aquatic mammals (e.g., dolphin) grouped in the fish cluster and the instance of newt classified with the reptile cluster, are no more common in the untyped data than in the typed data.

A hierarchical version of the ART network, HART [Bartfai 1995], uses the Zoo data to produce a hierarchical classification as an illustration of its abilities. The SCENT classification can be compared with the results presented in the HART paper. Both networks clearly become increasingly specific at lower levels and in that sense are *hierarchical*, however, there is no evidence of super-ordinate classification being developed within the HART tree structure, consequently the categorisation classes do not reflect natural groupings. The lower level classes of the SCENT model, do however often give natural sub-groupings, because of the guiding super-ordinate organisation developed during the tree's evolution.

SCENT's classification has also been compared to that of a traditional agglomerative clusterer [Everit 1993]. The traditional clusterer , of course, produces a deep, and unbalanced tree (figure 10). The depth of the tree is indicative of the sequential decision, and structuring, mechanism employed by the clusterer. Once a match on distance has been made there is no scope for restructuring the position of the decision nodes; this forces the tree's granularity to be fine grained and limits the production of generalised clusters such as those seen in SCENT. Another problem with this type of method is that no indication of the number of clusters in the data is induced, since all vectors are forced into a binary, similarity relation with a neighbouring vector or cluster.

The dendrogram, cannot identify the three major groupings that SCENT puts at level one. However by level three, four major groupings are apparent: fish, mammals, birds and others. The immediate subdivision of the mammal cluster, broadly into predatory and non-predatory, identified by SCENT, only emerges, less usefully, at the very bottom of the dendrogram.

Table 4 shows the category prototypes produced by SCENT for the mixed middle cluster of figure 9 (referred to as class B). The top level classification is almost all defined by the attributes no-hair, no-milk, has-eggs, has-backbone, has-tail. This prototypes more than one taxonomic group. Sub-categories are specified more precisely. For example sub-classes B1 and B3 (representing all the instances of birds) are prototypically all of the above attributes together with feathered, toothed, breathing, not-venomous and not-finned whilst B2 (representing the instances of fish) has all the B-class features plus not-feathered, not-airbourne, not-breathing, not-legged, aquatic, toothed and finned.

The superordinate prototypes produced by SCENT do not rigidly fix the sub-ordinate prototype, that is, a sub-ordinate category may contain a "don't care" item where its superordinate category has indicated a definite value of this attribute. This effect may arise from the stochastic process of node creation within the model, and it can provide flexibility in the classification of potentially anomalous data[1].

[1] It should be noted that the prototypes developed by either the SCENT or HART models are influenced by the semantic labeling present in the data set. Anomalous labeling within data vectors must

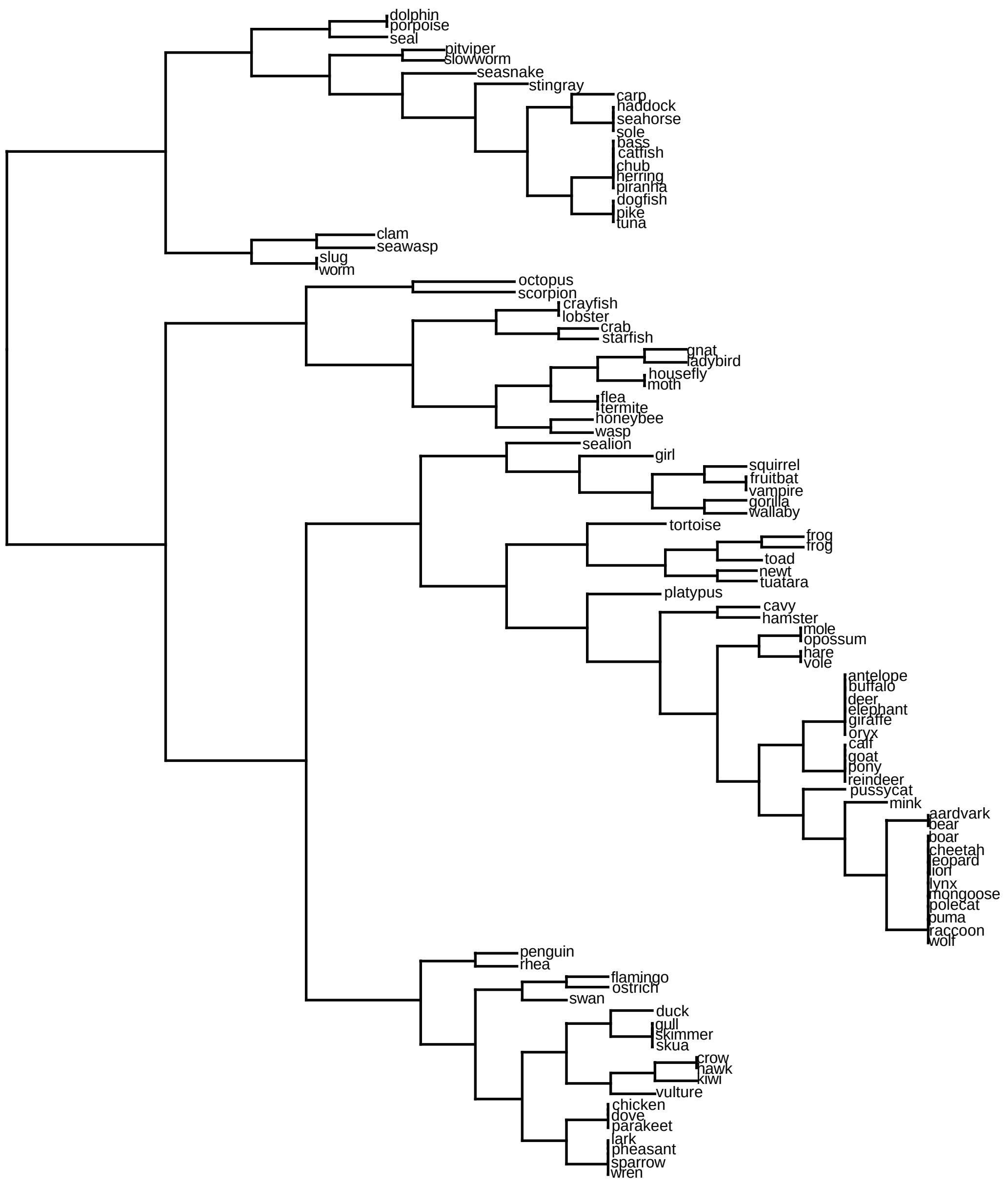


Figure 10. A dendrogram of the Zoo data set, produced by agglomerative clustering .

The bottom-up development of hierarchical models, such as HART, forces the propagation of any mis-identification of critical features through to the developed prototypes. Models which follow a top down approach to classification, such as SCENT, do not suffer from this problem. The initial selection of critical features is made using the variance of the entire data set, thus producing a coarse grained measure of classification. Subsequent selection of critical features is made on progressively smaller sets of input vectors (as patterns are classified, fewer need classifying), the classification therefore becomes more fine grained or focused as the tree evolves.

therefore be considered when viewing these results, especially against biological taxonomic classification.

Feature	Label	Class B	SubClass B 1	SubClass B 2	SubClass B 3	SubClass B 4	SubClass B 5
2	Hair	no	no	no	no	*	no
3	Feathers	*	yes	no	yes	*	no
4	Eggs	yes	yes	yes	yes	yes	*
5	Milk	no	no	no	no	no	no
6	Airbourne	*	*	no	*	*	no
7	Aquatic	*	*	yes	*	*	*
8	Predator	*	no	*	yes	*	yes
9	Toothed	*	no	yes	no	*	yes
10	Backbone	yes	yes	yes	yes	*	yes
11	Breathes	*	yes	no	yes	*	*
12	Venomous	*	no	*	no	*	*
13	Fins	*	no	yes	no	no	*
14	Legs	*	*	no	*	*	*
15	Tail	yes	yes	yes	yes	*	yes
16	Domestic	*	*	*	no	*	no
17	Catsize	*	*	*	*	*	no

Table 4. Category Prototypes for middle cluster of figure 9, the Zoo Tree. “Yes” indicates attribute presence (weight value >=1.0). “No” indicates absence of attribute (weight value <=0.0). “*” indicates a don’t care value for the attribute (1.0> weight value >0.0)

4.2.2.3 Comparison of Structure

The structural features of the resulting SCENT trees (table 5) indicate variation in the overall structure of trees produced in different runs, however this variance is not excessive.

Data Source	Leaf Nodes	Av. Branching	Av. Depth
Pictures	30	3.2	3
	38	3.5	3.2
	34	3.2	3.2
	27	3.2	2.7
	Average 32	Average 3.27	Average 3.02
Zoo Typed	30	2.53	3.57
	23	3.30	2.65
	24	2.77	3.58
	30	2.56	3.77
	Average 27	Average 2.79	Average 3.39
Zoo Untyped	29	3.00	3.03
	32	3.07	3.25
	32	2.94	3.00
	24	3.18	2.83
	Average 29	Average 3.05	Average 3.02

Table 5. Summary of the tree structures produced by SCENT over the three data sets and four runs.

Overall the trees produced for the picture data set were larger than those produced for either of the zoo sets. This is accounted for exclusively by these trees having a greater branching factor; indeed the zoo data tended to produce slightly deeper trees. In view of the model's growth criterion for new clusters, downwards for dense data points and sideways for spatially separated data, this implies that the picture data has greater spatial separation.

The two zoo data sets produced trees of slightly different shape. The removal of the type label caused the trees to be shallower but with more branches, which as before shows the untyped data to be the more spatially separated. The reason for this is that two similar vectors with identical type fields are slightly less similar when the type field is removed.

5 Conclusions

Using neural nets to perform data exploration, and thereby discover any structural knowledge implicit within the data is difficult; most models require the pre-imposition of a maximum number of clusters, and will normally classify the data to utilise all classificatory units. Some recent architectures have attempted to dynamically create tree structures to overcome the need for prescribing cluster number and to give a hierarchical view of the data. Their use though has been problematic, with a tendency to great sensitivity to multiple initial parameterisation. SCENT attempts to overcome this problem by being completely autonomous. Both CENT and SCENT produce consistent tree structures across a range of data sets. SCENT exhibits more stability then CENT with hierarchical data due to low variance.

The SCENT model has produced a consistently good performance over all of the data sets presented so far. The SCENT performance has been better than that of the two original DNTN networks over almost all the tests. In test 3 the Li performance was excellent and outperformed the SCENT, but the Racz and Klotz performance was significantly worse than the SCENT model. The SCENT performance has been good over both uniform data sets, where the Racz and Klotz network performs poorly, and data sets with ranges of cluster densities and sizes, that cause the Li network to perform poorly. The consistency of the SCENT performance over all types of data is notable and shows that the methods used within the model to assess each data set are both thorough and flexible.

The fact that the SCENT network has been able to produce performances that are close in terms of SSE to the Neural Gas network is very encouraging. It is harder for a dynamic network, particularly a tree structured network, to produce a good SSE result than it is for a flat static network such as the Neural Gas. The ability of the SCENT to provide the advantages given by the tree structure whilst maintaining good SSE results is a positive result.

From the results and analysis presented in section 4.2 it can be seen that the SCENT model is capable of discovering interesting hierarchical structure in unlabelled data, as well as providing a straightforward codevector reduction. The SCENT model has succeeded in producing a hierarchical tree structure reflecting the clear visual differences between the images and is capable of learning stable hierarchical clusters that include both super and sub categories from semantic data.

The development of classification hierarchy models such as HART and SCENT allows for rapid high level analysis of data sets. Critical features of the data are easily identified, as are any features of little statistical significance, at a number of levels within any natural hierarchy of the data set. The top-down approach of SCENT allows for the development of flexible prototype clusters based on hierarchical code vectors of an unlabelled data set.

The full theoretical relationship between training data, in general, and the resulting structures produced by SCENT is an issue currently being investigated.

References

Adams, R. G., Davey, N. & George, S. G. (1998). Analysing Hierararhical Data Using a Stochastic Evolutionary Neural Tree. *Proceedings of the International Symposium on the Engineering of Intelligent Systems* (EIS98), pp 268-275

Bartfai, G. (1995). An ART-based Modular Architecture for Learning Hierarchical Clusterings, *Neurocomputing*, 1995.

Bezdek, J.C., and Pal, N.R. (1995). Two Soft Relatives of Learning Vector Quantization. *Neural networks*, Vol.8, No. 5, pp. 729-743.

Butchart, K., Davey, N., Adams, R (1995). A Comparative Study of Three Neural Networks that use Soft Competition , proceedings of IWANN'95, 308-314.

Butchart, K., Davey, N., Adams, R (1996a). A Comparative Study of two Self Organising and Structually Adaptive Dynamic NeuralTree Networks", pp 93-112 in *Neural Networks and their Applications* , J.G.Taylor Editor, John Wiley.

Butchart, K., Davey, N., Adams, R (1996b). Hierarchical Classification with a Stochastic Competitive Evolutionary Neural Tree. Proceedings of ICNN96, Vol. 2, pp 1372 - 1377.

Butchart, K. (1996c). Hierarchical Clustering Using a Dynamic Self Orgainising Neural Networks. *PhD Thesis*. University of Hertfordshire.

Butchart, K., Davey, N. & Adams, R. G. (1997). An Investigation into the performance and representations of a Stochastic Evolutionary Neural Tree. *Proceedings of the International Conference on the Applications of Neural Networks and Genetic Algorithms* (ICANNGA 97), Springer Verlag.

Carpenter G.A, Grossberg S. (1987) A Massively Parallel Architecture for a Self-organising Neural Pattern recognising machine, *Computer Vision, Graphic and Image Processing,* vol. 37, pp 54-115.

Everit B.S. Cluster Analysis. Edward Arnold , London, 1993.

Gale, T. (1997). Perception and Semantic Information in Human Object Recognition: a Neuropsychological and Connectionist study. *Ph.D. Thesis*, University of Hertfordshire, 1997

Hertz, J., Krogh, A., Palmer, R. G. (1991) An introduction to the theory of Neural Computation. Adddison Wesley.

Krishnaiah, P.R. and L.N. Kanal (1989). Classification, Pattern Recognition, and Reduction of Dimensionality. Handbook of Statistics, vol 2. Amsterdam: North Holland

Li, T., Tan , Y., Suen, S. and Fang, L (1992) A structurally adaptive neural tree for recognition of a large character set. In: *Proc. 11th IAPR International Joint Conference on Pattern Recognition,* vol II,pp187-190, 1992.

Li T., Fang L. Q-QLi K. (1993) Hierarchical classification and vector quantisation with neural trees. *Neurocomputing* , vol. 5, pp 119-139.

Li, T., Tang, Y. Y., Fang, L. Y. (1995). A Structure-Parameter-Adaptive (SPA) Neural Tree for the Recognition of Large Character Set. *Pattern recognition,* Vol. 28, No. 4, pp. 315-329.

Martinetz, T. , Berkovich, S. and Schulten , K. (1993) Neural-Gas Network for Vector Quantisation and its Application to Time-Series Prediction. *IEEE transactions on Neural Networks.*, vol. 44 (4).

Metropolis, N., Rosenbluth , A. W., Rosenbluth , M.N. ,Teller , A.A . and Teller , E. (1953) Equations of state calculations by fast computing machines. *Journal Chemical Physics* . vol. 21, pp. 1087-1091.

Murphy, P.M. and Aha, D. W. (1992). Repository of Machine Learning Databases, Technical Report, Department of Information and Computer Science, University of California, CA, 1992.

Pal, N., Bezdec, J. and Tsao, E. (1993). Generalised Clustering Networks and Kohonen's Self-Organising Scheme. *IEEE transactions on Neural Networks* , vol. 44 (4).

Racz, J. and Klotz, T. Knowledge Representation by dynamic competitive learning techniques. *SPIE Applications of Artificial Neural Networks II* , vol. 1469, pp. 778-783, 1991

Sneath, P.H., and Sokal, R.R. (1973). Numerical Taxonomy, the principles and Practice of Numerical Classification , W.H. Freeman and Company, San Francisco.

Song, H., Lee, S. (1998). A Self-Organising Neural Tree for Large-Set Pattern Classification. *IEEE Transactions on Neural Networks.* Vol. 9, No. 3, pp. 369-380.

Yair , E. , Zeger , K. and Gersho, A. (1992). Competitive Learning and Soft Competition for Vector Quantiser Design. *IEEE transactions on Signal Processing,* vol. 40 (2).