DIMENSION REDUCTION FOR LINEAR SEPARATION WITH CURVILINEAR DISTANCES

Jonathan Winkley, Ping Jiang & Alamgir Hossain

University of Bradford
School of Computing, Informatics and Media
Richmond Road, Bradford, West Yorkshire
United Kingdom
j.j.winkley@bradford.ac.uk, p.jiang@bradford.ac.uk, m.a.hossain1@bradford.ac.uk

Abstract: *Any high dimensional data in its original raw form may contain obviously classifiable clusters which are difficult to identify given the high-dimension representation. In reducing the dimensions it may be possible to perform a simple classification technique to extract this cluster information whilst retaining the overall topology of the data set. The supervised method presented here takes a high dimension data set consisting of multiple clusters and employs curvilinear distance as a relation between points, projecting in a lower dimension according to this relationship. This representation allows for linear separation of the non-separable high dimensional cluster data and the classification to a cluster of any successive unseen data point extracted from the same higher dimension.*

Keywords: *Dimension Reduction, Curvilinear Distances, Clustering, Classification, Linear Separation.*

1 Introduction

As problems become defined by greater amounts of data, dimensionality can increase. The high dimensionality of data can lead to complex non-linearly separable clusters developing in the higher dimension space which are not easily identified by simple classification methods. The data may in fact contain values in a dimension which allow it to be perfectly separated, however the influence of other dimensions may obscure the required classifiable relationships present in the data.

An issue arises in high dimension data when the distance which separates clusters is too small to enable differentiation, or the data within the clusters themselves is too sparse. The ability to detect outliers in sparse high dimension data is a commonly tackled problem; [1] notes that while in one dimension a data point may be an outlier, in another it may belong to a typical cluster's representation.

This "curse of dimensionality" is commonly referenced; the greater the amount of information (dimensions) the lesser the ability to make sense of the data. This is relevant when addressing the *k*-nearest neighbour function due to the fact that with a higher dimension, standard Euclidean distance functions lose their usefulness and so clustering with such methods becomes less accurate. There are 4 problems which relate to the "curse of dimensionality" and the increasing number of attributes [2 , 3 , 4]: Optimisation becomes difficult, relative distance between extreme points converges to 0 (discrimination between nearest and farthest neighbour becomes poor), dimensions become "noise" given that their relevance to the data may be little and some dimensions may even "exhibit correlations among each other" - thus becoming redundant.

A technique capable of separating clusters would result in datasets on which successive linear-based operations can be performed – such as a simple binary classifier to identify successive data points' membership to a cluster. The k-means and vector quantisation techniques produce clusters which are initially identified according to the arithmetic mean of a number of values, with new clusters being identified when a mean is deemed too far (in a Euclidean space) from any other clusters'. When the clusters in classifiable data have a non-evident relationship mathematically, these methods won't produce the required results as they rely on the inherent properties of a cluster being similar. In some non-linear data sets this is the case as a data point may belong to a cluster only because its nearest neighbour also belongs, not because it shares a common property with the overall cluster.

Neural network based classifiers can also be applied to non-linear high dimension data in order to identify the non-linear relationship and classify successive points according to the properties of the data set. The problem with such approaches is their need for trial-and-error in the choice of the number of layers, neurons and iterations. The end result may also be impractical if the determined "solution" has actually fallen in a local minimum, thereby producing false results and reducing the sensitivity of the classifier.

The technique presented here for reduction/separation retains the same structure for all problems where the input data contains classifiable clusters which in their current dimensional representation are not linearly separable. Such a method enables simple implementation for any data set given that the principles remain the same in all cases: the cluster data is first projected into a lower dimension space where it is easily separated, before a single layer neural network such as a perceptron is applied to classify the data sets and successive interpolated points in the new space. Experiments on the

non-linearly separable Fisher's Iris data and other such available sets have shown this combined dimension reduction and classification method to be accurate to a higher degree than with a supervised neural network approach alone.

In Section 2 we look at dimension reduction using this method with the intention of using the lower dimension data for classification of future data points. Section 3 provides numerous examples on commonly used data sets, with details of classification results on both high and low dimensions and use in neural networks for comparison. Section 4 summarises the results before the conclusion and possibilities for further work are discussed in section 5.

2 Dimension Reduction

As highlighted, the measurement of straight-line (Euclidean) distance within high dimension data is unreliable given its tendency to misrepresent true topology. Similarly, identifying correlations (based on a least-squares approach) in the data as in PCA [5 , 6] is not guaranteed to be accurate in non-linear high dimension data given that such a distance relation is unreliable. When reducing dimensions, it is therefore feasible to assume that reconstructing data purely with such a distance is inadequate and will result in similarly inseparable clusters as in the higher dimension space. The Curvilinear Distance Analysis (CDA) [7] is considered here, yet extended to incorporate a method of separating clusters while reducing the intrinsic dimension of the data. CDA is able to preserve the higher dimension space's topology while also sufficiently reproducing the local environment. In the interpolation of successive data points, the classification to a cluster is based on the nearest-neighbour metric: where the single closest prototype determines where that new point will be positioned in the lower dimension space.

A scheme is proposed which takes as its input a multi-dimensional training data set, in which a combination of dimensional values signifies classification to a cluster. Let $X = [x_1, x_2, \cdots, x_M]^T$ be the data set of observations, where $x_i \in R^N$ is in an N dimensional space. Without loss of generality, we know that there are two classes, $X_A \in R^{M1\times N}$ and $X_B \in R^{M2\times N}$ in $X \in R^{M\times N}$, where $M=M_1+M_2$. Because X_A and X_B are not linearly separable, a nonlinear boundary has to be determined by multiple layered neural networks or nonlinear kernels if the classification is carried out directly in the high dimension. This can be a tedious trial-and-error process. In this paper, a dimensional reduction technique is applied for data pre-processing.

Processing each of the class clusters in turn, those points which typify the topology of the cluster are identified as "prototypes". These prototypes are then interlinked according to neighbourhood before each cluster is then connected to another via a single link and a graph created detailing curvilinear distances between all prototypes in the data set. The set $X = [X_A, X_B]^T \in R^{M\times N}$ is projected to a lower dimension $[X_a, X_b]^T \in R^{M\times n}, n < N$, where these distances are recreated as Euclidean, thus "flattening" the high dimension data and linearly separating the clusters X_a and X_b in the lower dimensional space. Based on distance to their closest prototype, successive points can be interpolated and projected from the high to the low dimension and once separated a simple classifier, e.g. a single layer perceptron, can be used to identify their parent cluster. The proposed approach is summarised in fig.1.

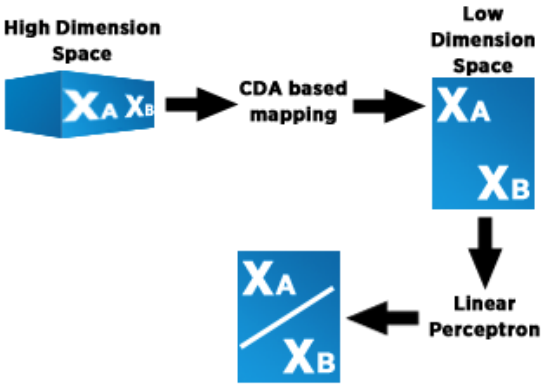

Fig.1 Dimension reduction with subsequent classification

The CDA method was itself an extension to the original CCA [8] and serves to unfold the data from high-dimension - *n*-space to the low-dimension *p*-space. The distances between prototypes of the single manifold are kept, whilst the remaining data points from within that manifold are projected such that their *p*-space distance is comparable to the *n*-space distance about the closest prototype. The principle is used extensively to reduce single 3D manifolds to 2D, in reduction of dimensions of visual data [9] and in Mass Spectrometry [10].

Fig.2 shows a simple, general example of how in 2D, the data clusters (letters, in this case) are more easily identifiable than in the 3D representation. If the original 3D data were to be used in a supervised learning scheme for classification, the properties of each cluster would be unclear as some overlap between the letters occurs. In such cases where this sort of data has been used for training, subsequent inputs can be incorrectly classified when their properties fit in the overlap. If the dimensionally reduced data is used there is a higher chance of correct classification given that during training the separate cluster sets are able to be identified as such, with each having significantly different properties to another.

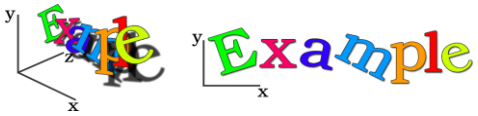

Fig.2 Reduction from 3D to 2D shows better separation

The documented CDA method is not equipped to tackle such a problem as defined here. If data consisting of multiple clusters is treated as a single combined manifold, the properties which make a cluster unique can be lost in its projection to a lower dimension. Treating the clusters as separate homogenous manifolds and then linking (chain-like) together as one before projecting enables internal cluster structure to be retained in the same way that the overall topology is in the conventional CDA.

As in CDA, the distance between prototypes of a cluster and prototypes in different clusters is computed using a routing algorithm (Dijkstra) and is therefore the total distance traversed along the path which connects them. The distance is required to be such that it enables retention of global topology between dimensions, so use of a statistical distance measuring similarity within a cluster is for this application inappropriate as it could significantly alter the internal structure of the cluster.

Whilst in CDA the interpolation of successive unseen data points uses the common Euclidean distance to project locally (the shortcomings of which when used in a high dimension were discussed above), a curvilinear distance more adequately represents a relation between data points in clusters as it enables visible separation over long distances. In this scheme the projection of supplementary interpolated points is therefore computed according to curvilinear distance also, given that CDA's low-error Euclidean distance measure for local representation would be impossible to achieve once the distance between prototypes has been "stretched" for maximum separation.

The initial steps in this method follow those set forth by Lee et al. in [9]. This implementation is intended to require as little involvement with the user as possible, so there are few adjustable parameters. Currently within the method the most significant variable parameter is the tolerable loss of the vector quantisation; if the value is higher then the quantised points generalise the clusters to a higher degree and thus computation is lighter, yet the resultant representation after interpolation can be seen to be less accurate.

2.1 Normalisation and Prototypes

From experimentation it is found that raw, high-dimension data is not always conducive to computationally fast operations. A distance relation in the *n*-space can be in the order of thousands, which when summing for the error can place great strain on a typical system. Normalising all input data first will solve this problem and given that the result in *p*-space is used solely for clustering purposes, the normalised distance representation will readily suffice.

A prototype is a data point in the *n*-space which will serve as a marker in the *p*-space around which data points can be projected. Using vector quantisation, the best prototypes representing these data sets can be determined. We approach each cluster individually and create prototypes within them in order to provide a decent generalisation of that cluster's topology. Any vector quantisation method may be employed, where here we use a dynamic vector quantisation which uses a competitive update scheme to bring the prototypes to a more reliable representation of the data:

```
for each cluster
        max_distance is 0
        for all data points in cluster
                if distance between 2 points is greater than max_distance
                        distance becomes max_distance
                end
        end
        radius = max_distance × tolerable_loss
        prototype = []
        prototype_num = 0
        while iteration is acceptable or prototype_num continues to increase
                for all data points in cluster
                        for all prototypes
                                if data point is not within radius of prototype
                                        data point becomes a prototype
                                        prototype_num = prototype_num + 1
                                else
                                        move closest prototype within radius by an amount which
                                        decreases with every iteration
                                end
                        end
                end
        end
end
```

As the neighbourhood window propagates through the data a series of initial prototypes are defined. Each iteration of the process sees the prototypes become more and more representative of the data as they move towards the centre of the points within their respective radii.

The convergence to an optimum number of prototypes will typically occur early on in the process and it is at this iteration we can stop the vector quantisation, or choose to continue on for a set number of iterations. This step is

performed for each cluster, with all prototypes being held in a 3-dimensional data matrix (of size $|prototypes| \times |dimensions| \times |clusters|$).

2.2 Link Prototypes According to Neighbourhood

This step requires another initial neighbourhood radius to be set, which in this instance is the mean of the 3 shortest distances between all prototypes (to enable at least one point to be encompassed by the radius for the first iteration):

$$k_{init} = \frac{1}{3}\sum_{i=1}^{3} a_i \tag{1}$$

Where a is an ordered list (from low to high) of distances between all prototypes in the cluster. The step-increase of the neighbourhood per iteration is then found with the initial neighbourhood and the maximum distance between prototypes:

$$k_{step} = k_{init} \times \max[a] \tag{2}$$

A square linkage matrix (3) is also created which contains information about which prototypes within a cluster are connected. It consists of $|a|^2$ entries and is initially populated entirely by "Inf" (∞) values to symbolise all are initially non-traversable links.

$$\begin{bmatrix} Inf & \cdots & Inf \\ \vdots & \ddots & \vdots \\ Inf & \cdots & Inf \end{bmatrix} \tag{3}$$

When evaluating each individual prototype i, if another, j, falls within the neighbourhood radius then the linkage matrix updates to accommodate the distance between the two at that index point ($link_{ij}$). Once the neighbourhood is evaluated, Dijkstra's algorithm is employed to ascertain whether or not prototypes can now reach all other prototypes through the created linkages. In the initial instance with the first prototype this will obviously be impossible; we decide if all linkages are traversable by returning the maximum distance in the Dijkstra matrix: if the maximum distance remains "Inf" there are still some unreachable prototypes. If some are still unreachable, the neighbourhood radius increases by a step-size and the evaluation process continues until the maximum distance is reduced. Once the linkages are all traversable, it can be said that the cluster is fully connected. The process repeats for all clusters until they are all internally linked.

The linkage matrix at each stage updates with the first calculated distance between prototypes: if the neighbourhood window happens to include a prototype via Euclidean distance before it is reachable via curvilinear distance through Dijkstra, the Euclidean distance becomes the shortest from one prototype to another. However, if the curvilinear encompasses a prototype before the neighbourhood expands, then that distance is the shortest. This avoids a web-like connection matrix which can cause subsequent projection errors.

CDA originally used a *k*-nearest neighbours approach to internal linking which sometimes would result in a parasitic connection between parts of a manifold. This automated method is more suited to the CDA process than the documented original as it significantly reduces the possibility of such connections and greatly assists in this CDA adapted for cluster-separation.

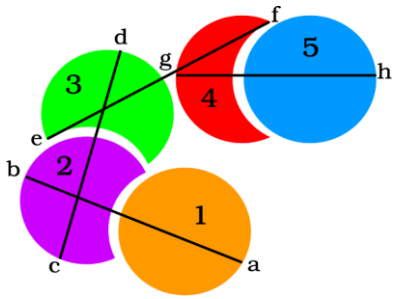

Fig.3 Linking of the clusters by maximum distance: 1 to 2 via ab, 2 to 3 via cd, 3 to 4 via ef and 4 to 5 via gh.

2.3 Cluster Linking for Unfolding

In order to represent the clusters in a lower (unfolded) dimension such that the topology of the higher dimension is retained – yet maximum separation achieved – we link the clusters through their furthest distance from each other. That is to say that each cluster, once internally linked, has each of its prototypes evaluated for its Euclidean distance to all other prototypes in other clusters. The maximum distance from one cluster to another (i.e. furthest distance between two prototypes of different clusters) then becomes the minimum linking distance and therefore the shortest distance present between clusters in the lower dimension. Fig.3 demonstrates the linking, with clusters 1 through 5 linked independently to their closest neighbour forming a chain link from the first to the last. It would also be possible to link clusters by their closest points, but for this application maximum visible separation of clusters is desired and maximum distance in the high-dimension representation is the most appropriate whilst retaining structural properties of the clusters.

2.4 CDA Projection

The determining of the curvilinear distances between all prototypes in all clusters occurs as in the original CDA implementation, using Dijkstra's algorithm to provide a matrix which contains all pairwise curvilinear distances. The projection of these prototypes also follows the typical CDA methodology which is covered in more detail in the original authors' work [7 , 9 , 11] with some further information in [8] covering the choice of some of the variable parameters.

The projection involves reducing the error between the computed curvilinear distances of the *n*-space and the new equivalent Euclidean distances in the lower dimension *p*-space. The result is an "unfolded" or "flattened" representation of the original *n*-space data which retains the distances between prototypes of a cluster yet separation between clusters has been maximised.

2.5 Interpolation of Supplementary Points

As put forth by the authors in [7 , 9], once the prototypes have been located in the lower dimension, the data which these prototypes typify must be similarly placed also. Given that the prototypes represent the data's clusters, their projections in the *p*-space can be taken as landmarks around which to project other points within the data set.

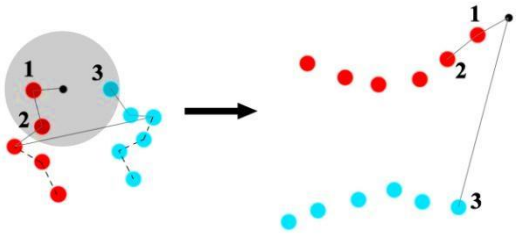

Fig.4 Selection of 3 closest prototypes to point (smaller, black point; light grey neighbourhood) in higher dimension and the subsequent projections in the lower dimension

Given that the linkage between clusters is a single connection, the resultant representation in *p*-space is somewhat stretched in order to maintain the curvilinear distances calculated. This then forces the original interpolation scheme developed for CDA to be further modified to accommodate the change. Therefore the "local", direct mapping of point-to-prototype Euclidean distance is replaced with a "global" mapping using curvilinear distances (which may be larger than Euclidean) - as was used in the projection of the original prototypes. This means that instead of locating closest prototypes to a data point and projecting in *p*-space according to Euclidean distance to them (as in the original CDA), we first identify the 3 closest prototypes according to their Euclidean distance. Had just 1 been taken as reference, the projection of the point may be anywhere 360º about the prototype; 2 prototypes would result in 2 possible positions whereas 3 allows for adequate triangulation of the interpolated point.

We determine the first closest prototype's distance to the other two not through the straight-line distance but the curvilinear, as it may be true that all prototypes belong to different clusters and therefore to represent locally (given the stretching of the links) is not possible. In this way, we can say that this modified CDA works such like a nearest-neighbour classifier and the projection position in the lower dimension is identified based on the single closest prototype.

The optimization of the error function works the same as with CDA, whereby a single point is moved in relation to all others in order to minimize the error between the high dimension curvilinear and the low dimension Euclidean distances to the closest prototypes. Once the original data set is projected, clusters are visible in the *p*-space as being linearly separable: so long as the maximum distance between clusters is large enough to overcome possible projection errors which may project prototypes in too close a proximity to others from different clusters.

It can be proven that a data set can always be made linearly separable with the proposed method if the distances from the high dimension space are retained; thus a single layer perceptron is sufficient for subsequent classification.

Assume X_A and X_B are mapped to a lower space to be X_a and X_b. It is theorised that X_a and X_b are linearly separable if the linking distance $d > diam(X_a) + diam(X_b)$, where $diam()$ is the diameter of a cluster, i.e. maximum in-cluster distance in (3).

Let the furthest points for X_A and X_B be x_A and x_B in the high dimension. They are projected to the low dimension space as x_a and x_b. A perceptron can be constructed with weight and bias:

$$W = \frac{(x_b - x_a)}{\|x_b - x_a\|} \tag{4}$$

$$b = -x_a \cdot \frac{x_b - x_a}{\|x_b - x_a\|} - diam(X_a) \tag{5}$$

The line function for classification can be written as:

$$net = W \cdot x_i + b = \frac{x_b - x_a}{\|x_b - x_a\|}(x_i - x_a) - diam(X_a) \tag{6}$$

With the perceptron output $f(x) = sgn(net)$. For any sample x_{ai} taken from cluster *A* the net output can be obtained from (6):

$$net(x_{ai}) = \frac{x_b - x_a}{\|x_b - x_a\|}(x_{ai} - x_a) - diam(X_a) \leq \|x_{ai} - x_a\| - diam(X_A) < 0 \tag{7}$$

Therefore the perceptron output $f(x_{ai}) = 0$.
For any sample x_{bi} taken from cluster *B*, the net output can again be obtained from (6):

$$\begin{aligned} net(x_{bi}) &= \frac{x_b - x_a}{\|x_b - x_a\|}(x_{bi} - x_a) - diam(X_a) \\ &= \frac{x_b - x_a}{\|x_b - x_a\|}(x_{bi} - x_b + x_b - x_a) - diam(X_a) \\ &= \frac{x_b - x_a}{\|x_b - x_a\|}(x_{bi} - x_b) + \|x_b - x_a\| - diam(X_a) \end{aligned} \tag{8}$$

As $d = \|x_b - x_a\| > diam(X_a) + diam(X_b)$:

$$\begin{aligned} net(x_{bi}) &= \frac{x_b - x_a}{\|x_b - x_a\|}(x_{bi} - x_b) + diam(X_b) \geq diam(X_b) - \frac{\|x_b - x_a\|}{\|x_b - x_a\|}\|x_{bi} - x_b\| \\ &= diam(X_b) - \|x_{bi} - x_b\| \geq 0 \end{aligned} \tag{9}$$

Therefore the perceptron output $f(x_{bi}) = 1$. The projection in the low dimension space is linearly separable.

3 Common Data Set Examples

Many data sets exist for the testing of classification methods and dimension reduction techniques. Commonly the most difficult problem is in representing and separating non-linearly separable data. The publically available data sets used here are Fisher's Iris Data, Bupa Liver Data, Breast Cancer Data and Wine Data – all of which are available at the UCI Machine Learning Repository [12]. A 3 dimensional set of artificial test data was created to aid in the visualisation of the reduction method given that the other sets are impossible to project (visualise) in their original high dimensional form.

All data sequences are split into two separate sets, with one set used in the prototyping and the other for successive testing. The dimensionality of each data set varied significantly, with the maximum being 13 for the Wine Data and the minimum being the 3D artificial data.

The results of each reduction can be seen in fig.5, where the resultant projection of successive data is promising yet obviously not without error (discussed in the next section). The prototypes of the test data are projected as block-colour circles, with the interpolated points from both the initial test sequence and the successive data being distinguishable by their darker edges.

Using the curvilinear method we optimize placement to the point where the overall error between recreated distance in the lower and actual distance in the higher dimension is reduced to less than 0.1. In practice this results in reasonable representations of the original data, whilst not being too computationally time consuming. If the application requires it, the maximum number of iterations of error reduction can be set – however to achieve optimal representation this would require other parameters within the CDA method to be adjusted to accommodate the change.

It can hopefully be clearly seen that the each of the clusters within the original data are reproduced separately in the lower dimension, and the interpolated points also fit within the respective boundaries laid out by their prototypes.

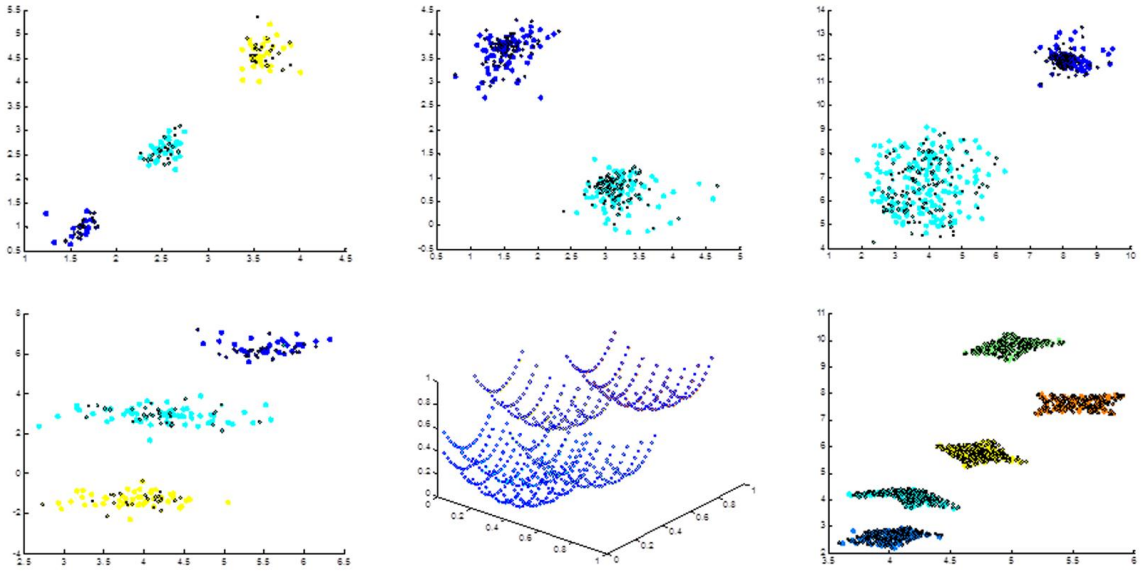

Fig.5 (Top) Fisher's Iris Data, Bupa Liver Data, Breast Cancer Data. (Bottom) Wine Data, 3D square-bowl data in raw form before the results of its dimension reduction.

4 Testing and Results

Whilst it is evident that the clusters are adequately represented as linearly separate in the lower dimension, it is also possible to see that in some data sets there will be miss-classifications of some successive data points. Given that the neighbourhood radius is used to determine the membership to a cluster, if a prototype falls too close to a point which does not belong to that cluster, the point immediately associates itself with that cluster. This is evident in the Wine Data set where one point which would normally belong to the middle class has in fact been identified as being a member of the top class.

What should be noted is that within Fisher's Iris Data, the two non-linearly separable clusters of Iris Virginica and Versicolor have been separated and the interpolated points have been successfully tagged to the correct cluster. With many supervised learning techniques this would also be possible, however through prototyping the initial clusters we have achieved a generalised view of each set, to which the neighbourhood function of successive points adequately allows for correct interpolation.

Using Matlab the effectiveness of the dimension reduction/clustering can be tested using a comparable neural network approach. A feed-forward back-propagation network is capable of solving almost any problem provided the network parameters are correctly chosen. In this test the Iris data in its raw form is submitted to the network and training commenced. The network consists of 2 layers and 20 neurons in the input layer and is trained for 1000 epochs or until the generalisation stops improving. Then the pre-processed, dimension reduced data is submitted to a similarly constructed 2 layer network.

With the non-reduced data, the classifications are returned as values which must be taken to one significant figure to provide a class; the reduced data provides correct classes of 1, 2 or 3 straight from the network. In the case of the non-reduced data 97% were correctly classified after hard limiting the output to an integer class. With the raw data 100% of Irises were returned as the correct class with no further processing required. The difference in performance error was 10^{-9} less for the reduced data, producing a more accurate result.

Conducting an experiment with a 3 layer network with different numbers of hidden layer neurons also returns a similar result. With 10 neurons in the hidden layer, the maximum error in classification for raw data is -1, where the maximum for the processed is 10^{-8}. With 20 neurons, the raw data training error increases to -2 at the point where training reaches the best performance; processed data again has a maximum classification error of 10^{-8}. These results show that the networks are much more receptive for classification if the data being classified can first be separated.

Given that the data can be separated in the lower dimension space with this method, it is now suited to simple binary classification; one perceptron could be used for each cluster whereby the output would be a 1 or 0 depending on its membership to that cluster. Using the high dimension, non-linearly separable data, the perceptron training fails to converge and therefore cannot be used as a correct classifier. In Matlab three perceptrons are trained with the raw data and the processed data, to a maximum of 1000 epochs each or earlier if the performance reaches the required value. Fig.6 shows the training performances, with the Virginica and Versicolor sets in the high-dimension failing to achieve a suitable weight and bias value allowing for generalization. In the lower dimension, all sets are correctly trained, with results reflecting this. Presenting values to the higher-dimension-trained networks gave an error of 60% for the two conflicting sets, where the lower-dimension-trained networks had a 0% error rate.

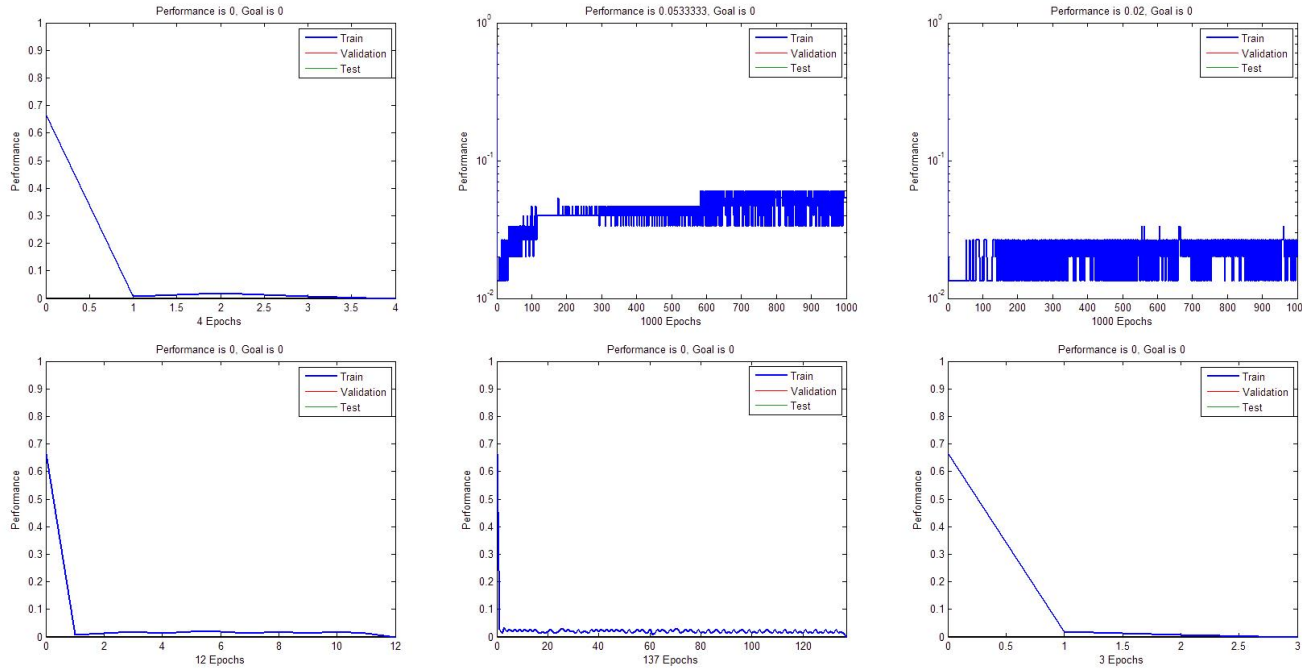

Fig.6 (Top) Training of perceptrons with raw iris data. (Bottom) Training of perceptrons with processed data. It is evident that the training of the raw data is ineffective in comparison.

4 Conclusion

This curvilinear separation and dimension reduction method consistently provides results which fully satisfy the criteria needed for interpolation and classification of unseen data points. In the dimension reduction are accomplished two feats: production of a data set which retains its general topology from the high to the low dimension (with an allowable error) - thereby maintaining overall data set characteristics - and an increased usability of the set for successive operations with linear classification methods. Its speed of interpolation of data points would be desirable in many other methods, provided that the prototype data has adequately summarised the current cluster topology.

It is in its offline training a time consuming process. With high-dimensional data consisting of an extensive amount of data points, the vector quantisation and linking steps can take copious amounts of computer operation cycles, yet it is in the reduction of projection error of the prototypes that we find the lengthiest process. Given a vast amount of data, linking the prototypes which generalise the component clusters results in an expansive graph which must be evaluated in every time-step during the minimisation of the projection error. When the projection space is of multiple orders of 10 larger than the initial data space (due to the "flattening" of the manifolds in the reduction method), the random projections which are to be relocated to minimise error can require that great distances be traversed in order to reach their optimum placement. In these tests the 5 cluster artificial data took longest to optimise placements to a highly accurate degree. Yet once the prototypes of the training data are located, successive interpolation of unseen points can be done in real-time in a matter of cycles to a very high degree of accuracy. With this speedy interpolation being followed by a simple perceptron, the combined system is both fast and accurate in its classification of unseen data points from a non-linear high dimension data set.

The only intended variable parameter in this system is the tolerable loss of the vector quantization step. Setting the value to closer to 1 will result in fewer prototypes of the data and therefore faster projections yet the trade-off is a lower accuracy result. With a lower tolerable loss (examples here used 0.1) the reproduction maintains obvious shape characteristics of the clusters given the increased number of prototypes. Ultimately it is at the discretion of the user of the method to decide on the accuracy of the topology reproduction given its trade-off with training speed.

The interpolation technique places the successive points in the lower dimension space within a few short iterations, and as such once the data's clusters have been prototyped it is possible to use the system in a real-time application to determine membership to a state. A problem may arise if the prototypes of the data begin to generalise only a small subset of a cluster. As the system evolves, more data may be present in a cluster which causes them to become sparser and as a result the prototypes don't generalise that cluster as well. Therefore during interpolation, closeness to the correct state can be reduced. Re-evaluating the system's prototypes periodically will overcome this issue, and a further enhancement could be achieved through ensuring that prototypes of the data also encompass boundaries of the clusters themselves.

At present, the nearest prototype to a data point determines the state to which that point belongs. It is supposed that if prototypes have a strength of belonging to a cluster, the closest prototype can be evaluated to see how likely it is that the new point belongs to it. If the distance to the next prototype is further, yet the strength of belonging to that cluster is greater, a trade-off mechanism can be evaluated to more adequately classify the data point to a cluster. It is in the adjusting of this neighbourhood function that the method will become a more useful tool in the supervised classification of data sets and the successive unsupervised interpolation of unseen data points.

References:

[1] Aggarwal, C. C. and Yu, P. S. "Outlier detection for high dimensional data," in in 'ACM SIGMOD International Conference on Management of Data', ACM: Press, 2001, pp. 37--46.

[2] Kriegel, H.P. and Zimek, A. "Detecting clusters in moderate-to-high dimensional data: subspace clustering, pattern-based clustering, and correlation clustering," Proc. VLDB Endow., vol. 1, no.2, pp. 1528-1529, 2008.

[3] Beyer, K., Goldstein, J., Ramakrishnan, R. and Shaft, U. "When is "nearest neighbor" meaningful," in In Proceedings of the 7th International Conference on Database Theory (ICDT, 1999).

[4] Hinneburg, A. Aggarwal, C. C. and Keim, D. A. "What is the nearest neighbor in high dimensional spaces," in In Proceedings of the 26th International Conference on Very Large Data Bases (VLDB, 2000).

[5] Pearson, K. "On lines and planes of closest fit to systems of points in space" Philosophical Magazine, vol. 2, no.6, pp. 559-572, 1901.

[6] Miranda, A. A., Yann A., Borgne, l. and Bontempi, G. "New Routes from Minimal Approximation Error to Principal Components," Neural Process. Lett., vol. 27, no.3, pp. 197-207, 2008.

[7] Lee, J. A., Lendasse, A. and Verleysen, M. "A robust nonlinear projection method," in In Proceedings of ESANN'2000, Belgium, 2000, pp. 13--20.

[8] Demartines, P. and Herault, J. "Curvilinear component analysis: a self-organizing neural network for nonlinear mapping of data sets," Neural Networks, IEEE Transactions on, vol. 8, no.1, pp. 148-154, 1997.

[9] Lee, J. A., Lendasse, A. and Verleysen, M. "Nonlinear projection with curvilinear distances: Isomap versus curvilinear distance analysis," Neurocomputing, vol. 57, pp. 49-76, 2004.

[10] Katajamaa, M., Miettinen, J., and Orešič, M. "MZmine: toolbox for processing and visualization of mass spectrometry based molecular profile data," Bioinformatics, vol. 22, no.5, pp. 634-636, 2006.

[11] Lee, J. A. and Verleysen, M. "Curvilinear distance analysis versus isomap.," 2004, pp. 49--76.