Preserving Class Discriminatory Information by Context-sensitive Intra-class Clustering Algorithm

Yingwei Yu, Ricardo Gutierrez-Osuna, and Yoonsuck Choe

Department of Computer Science

Texas A&M University

Technical Report: TR 2005-2-3

Email: {yingweiy,rgutier,choe}@cs.tamu.edu

February 25, 2005

Abstract

Many powerful techniques in supervised learning (e.g. linear discriminant analysis, LDA, and quadratic classifier) assume that data in each class have a single Gaussian distribution. In reality, data in the class of interest, i.e., the object class, could have non-Gaussian distributions and could be isolated into several subgroups by the data from other classes (the context classes). To address this problem, one possible way is to partition one class into several subclasses. This intra-class clustering should depend on the data structure of the class of interest (object class) as well as the distributions of all other classes (context classes). In this paper, we presented a novel method of intra-class clustering which can divide a non-Gaussian class data into several Gaussian-like clusters, and at the same time this algorithm is context sensitive, which can maximally reduce the overlapping among resulting classes and also between the object class and the context classes. The method can serve as a general

data preprocessing method to improve performance of supervised learning algorithms such as LDA and quadratic classifiers.

1 Introduction

In supervised learning, many effective methods have an assumption that the data has a Gaussian structure. For example, Fisher's linear discriminant analysis (LDA) (Duda et al. 2001) can find an optimal projection to discriminate data from different classes by utilizing the within-class and between-class covariance information of labeled examples. However, one of its limitations (Zhao 2000) is that if the data is non-Gaussian, its performance can be severely degraded. This kind of problem is also present in quadratic classifiers. Figure 1 shows a configuration of two classes of data (represented as white and black dots) in two-dimensional space. Because each class has a multi-modal distribution, the quadratic classifier gives poor classification accuracy on testing data.

In this paper, we want to solve such a problem by breaking a single object class into several subclasses. The basic idea is to separate the data in the class of interest (object class) into several single Gaussian-like clusters and at the same time considering the combined data of all the other classes (context classes) to provide boundary information for the partitioning. The separation of data into single Gaussian-like clusters can address the non-Gaussian distribution problem, while clustering based on context information can maximally reduce the overlapping between resulting subclasses.

The concept of intra-class clustering we introduce here is different from conventional clustering methods. Besides aiming to separate data in one class into multiple Gaussian-like subclasses, the partition method must also account for the context information provided by all the other classes, which may help isolate the object class into several disjoint groups. This aspect is best demonstrated in figure 2: these data points are generated from a single Gaussian distribution that is centered at $(0, 0)$, thus without presenting the context data, it

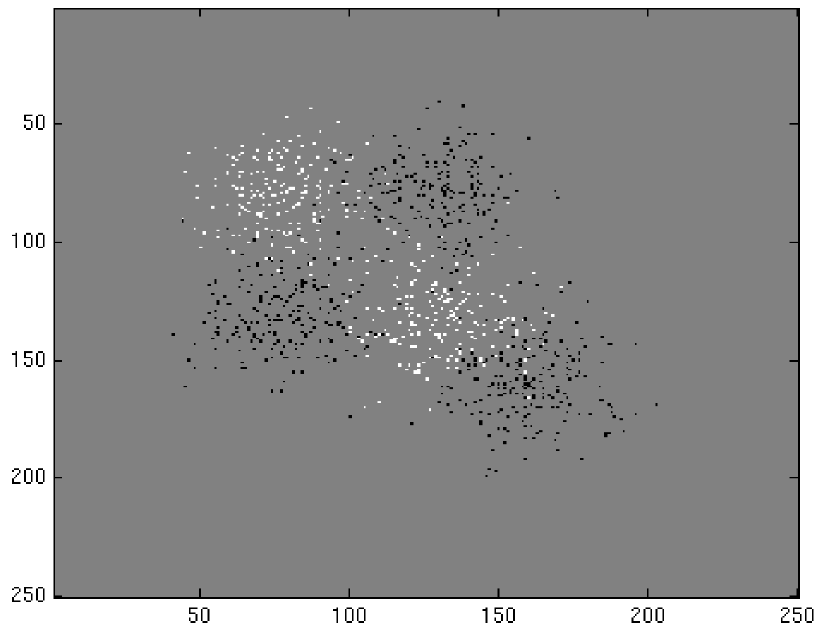

Figure 1: **Two-class data with multi-modal Gaussian distributions.** Examples in the two classes (black and white) are randomly generated from five Gaussian distributions. The quadratic classifier tends to treat each class as a single Gaussian distribution, and this assumption can yield poor performance.

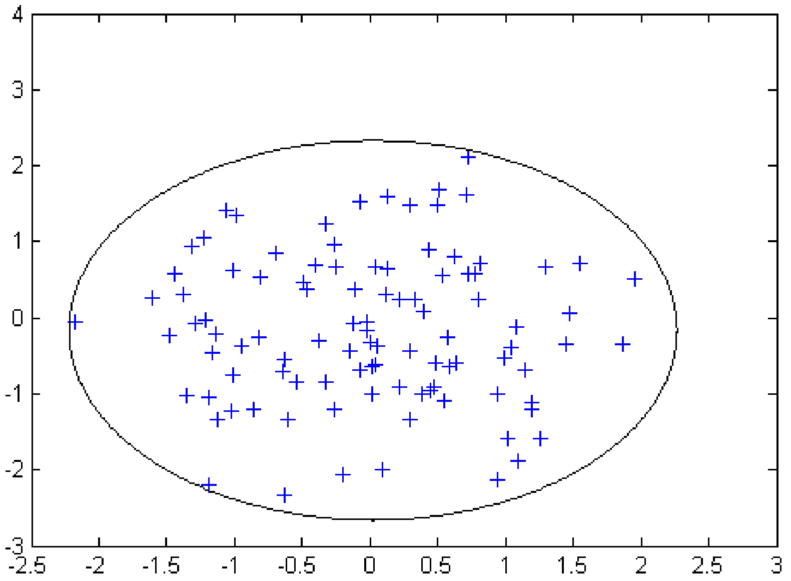

Figure 2: **Random data with Gaussian distribution.** The data points in the plot could be best matched with a single Gaussian centered at $(0, 0)$ where no context data are presented.

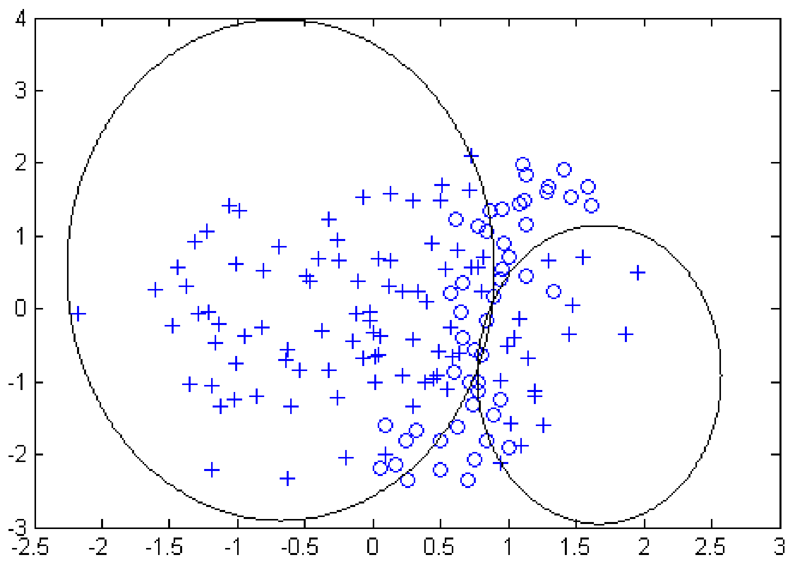

Figure 3: **Object class clustered into two subclasses in the presence of the context data.** The data in the object class represent by + are exactly the same as in figure 2, but with the presence of the context data which is represented by ∘, the object class should be separated into two clusters. This method removes the unnecessary overlap between the subclasses.

is best clustered into only one cluster as shown by the circle in the figure. However, with presence of the context, we need a different strategy for clustering. As shown in figure 3, the data marked + symbol is the same as in figure 2 but data points from another class (the context class) are intervening (marked ∘). With the intervention of the context, the object class naturally separates into two clusters: one on the left, and the other on the right of the context cluster, as indicated by the two ovals. This example shows how the context data can have a significant influence on the intra-class clustering results. The intra-class clustering algorithm in this paper will be specifically account of the influence by the context classes.

The rest of the paper is organized as follows. The next section introduces the method of intra-class classification, and followed by results from the proposed method on both synthetic data and real data. Finally, a brief discussion is presented, followed by conclusion.

2 Intra-class Clustering

It is easy for humans to visually cluster data even in the presence of context data (figure 3). The reason could be that we tend to utilize the low spatial frequency information in the visual scene. Thus in this sense, extracting low spatial frequency information could be helpful in guiding a clustering algorithm to perform intra-class clustering. Density estimation using convolution over the examples by Gaussian kernel method (Paren 1962) can extract such low spatial frequency information we need, so we start by obtaining such a density map which can reflect the object class data structure as well as the context data distributions.

2.1 Density Estimation of Object Class with Context Data

When there is only the object class, the data density can be estimated by using Gaussian kernel methods (Paren 1962). However, in case of estimating density of object class embedded in a certain context, the density of the context class data must also be considered as they can have impact on the object class structure. For context data, a few examples that are very sparsely distributed may not be able to help separate the object class, but if the density is high enough, it may isolate the object class to be isolated into several subclasses. In order to account for the context data's ability to partition the object class, based on their density and spatial distribution, we can use negative Gaussian kernels. For vector $x \in \Re^p$ (p dimensional real space), the density $d_i(x)$ created for class ω_i can be defined as:

$$d_i(x) = \sum_{x_e \in \omega_i} G(x, x_e, \Lambda) - \sum_{x_i \in \overline{\omega_i}} G(x, x_i, \Lambda), \tag{1}$$

where x_e is a sample from the object class, x_i that from the collection of context classes $\overline{\omega_i}$, Λ a constant covariance matrix for the Gaussian kernel G. The Gaussian centered at y in p dimensions is defined as:

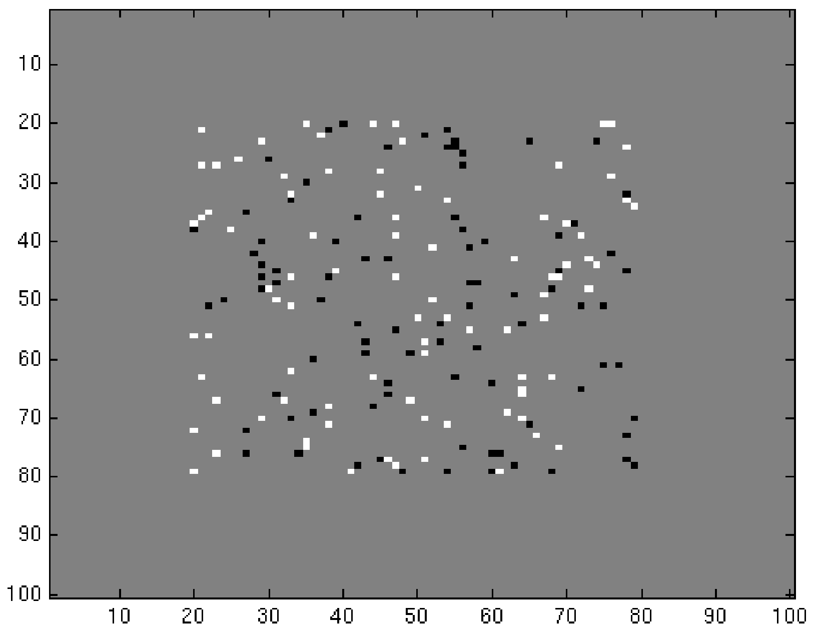

Figure 4: **Random data example in 2D.** Two classes (white and black) are randomly generated in a 2D plane. The white dots form the object class while the dark dots form the context class.

$$G(x, y, \Lambda) = \frac{1}{(2\pi \det(\Lambda))^{p/2}} \exp(-\frac{1}{2}(x-y)^T \Lambda^{-1} (x-y)). \tag{2}$$

For example, the density estimation of the white class (object class) in figure 4 is shown in figures 5 and 6.

The covariance matrix Λ is a user-adjustable parameter and the choice of it can be based on several existing techniques, such as the optimal method, or the robust method, etc. reviewed by Silverman (1986), and this matrix can be used to control the number of local maxima in the density map.

Once we have the density estimation with the context, we can use it as a map which guides each example to locate their local maximum. Then, grouping the examples from the object class with the same local maximum point can yield our desired result of intra-class clustering.

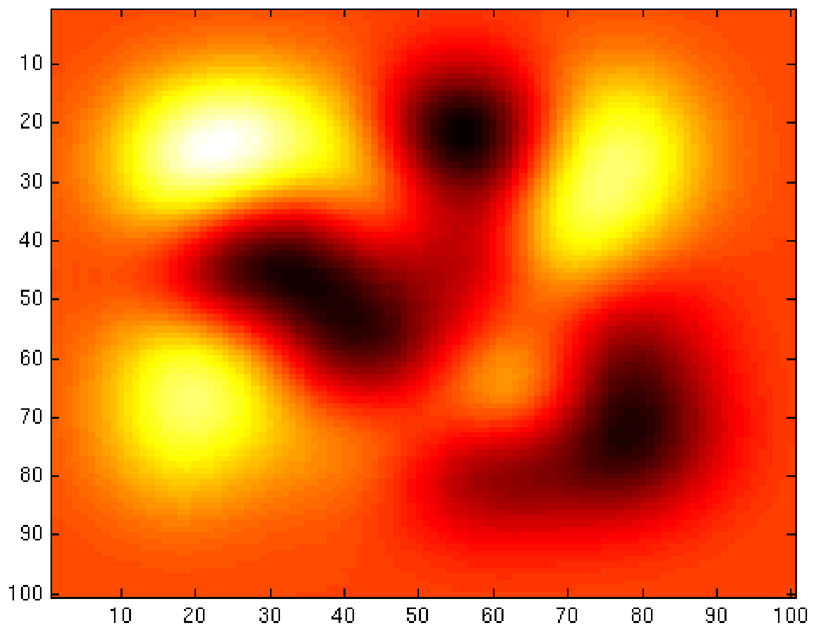

Figure 5: **Context sensitive density estimation result.** The white dots in figure 4 with high density are represented in light color, and the context distribution in dark.

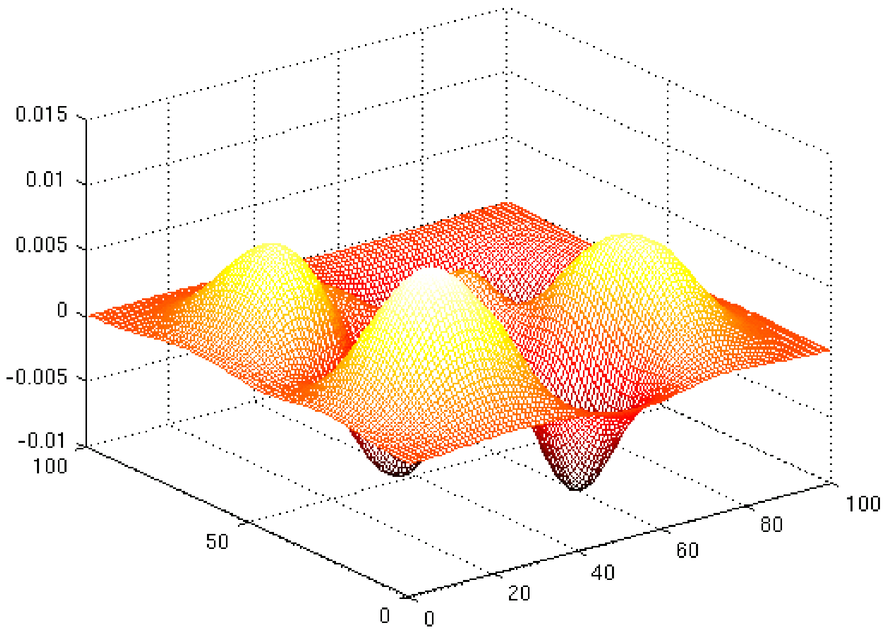

Figure 6: **Context sensitive density estimation result shown in 3D.**

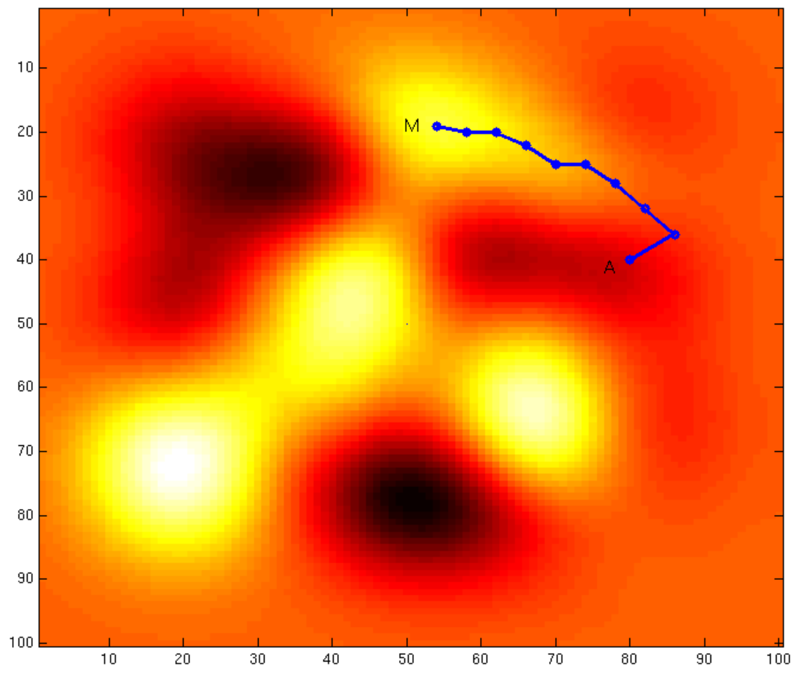

Figure 7: **Finding local maximum on a context sensitive density map.** From a point A, the steepest-ascent algorithm can find a local maximum point M for A. ($r = 5$)

2.2 Clustering by Finding Local Maximum

Because we use Gaussian functions as the kernel to estimate the density in equation 1, the first derivative of function d is continuous. This continuous property is suitable for density clustering algorithms (see, e.g., Kowalewski 1995; Hader and Hamprecht 2003) including ours, so that we can cluster the data in object class according to the function d. The basic idea of density clustering algorithm is to assign examples on the same "mountain" to one cluster. Hence, the data in each new cluster will share one local maximum, and that in a locally Gaussian-like distribution. To locate the local maximum, we can employ steepest-ascent algorithm, to be described next.

2.2.1 Steepest-Ascent Hill Climbing Algorithm

To search the local maximum for example x from the object class, we can use the gradient of the density map. The gradient of the density map for object class i is a vector ∇d_i that represents the magnitude and direction of the steepest slope.

The gradient can be calculated locally, and the position of the local maximum y can be found by iterating the formula

$$y \leftarrow y + \eta \nabla d_i, \tag{3}$$

where y is initialized to x, and η is the learning rate (see, e.g., Russell and Norvig 2003).

After finding the local maximum for all examples, then we can assign the examples sharing the same local maximum with equal labels (Hader and Hamprecht 2003). For clusters containing too few examples, say only one example, this could be because it is surrounded by too many context data and it is highly possible to be due to noise in the data. We can either remove this data point from the training set or assign it to a nearby cluster.

3 Experiments and Results

First we will test intra-class clustering algorithm in section 3.1. Then, we will use intra-class clustering in classification tasks, where an initial set of n classes are broken down into $m(> n)$ classes, and then combined back into n classes (section 3.2 and section 3.3).

3.1 Experiment 1: Intra-class clustering in 2D random data

We first applied our intra-class clustering algorithm on a two-class, randomly generated examples configured as in figure 4. The data in both class are uniformly distributed in a two dimensional space, and both their expected means are all located in the center of the plot.

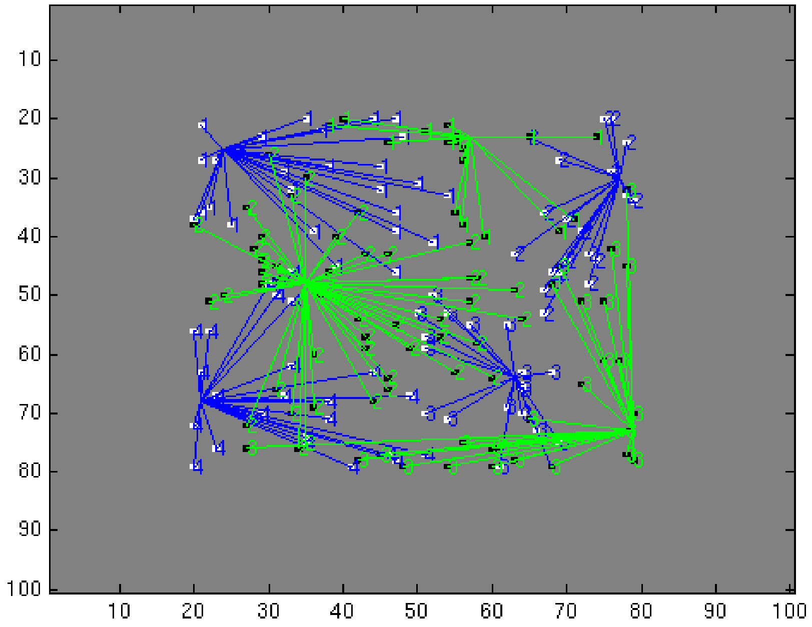

Figure 8: **Intra-class clustering result.** This figure is the intra-class clustering result of figure 4. The white dots are clustered into four groups and the dark dots are clustered into three groups.

The labeling result of intra-class clustering on this data is shown in figure 8. The white dots are clustered into four groups and the dark dots into three groups. Note that there is little overlap among the subgroups.

3.2 Experiment 2: Quadratic classifier with intra-class clustering

In the second experiment, we tested how the intra-class clustering method could be used as a preprocessing method to improve the quadratic classifier's accuracy. Two classes of random data are generated, as shown in figure 1. The white dots are generated by two Gaussians while the dark dots by three Gaussians. The quadratic classifier has significant disadvantage in this case because it tends to model the dark dots of all three Gaussian blobs into one single distribution and thus there will be a large overlap to class of the white dots. With intra-class clustering, the two classes can be divided into five groups of Gaussian distributions. The clustering results are shown in figure 9.

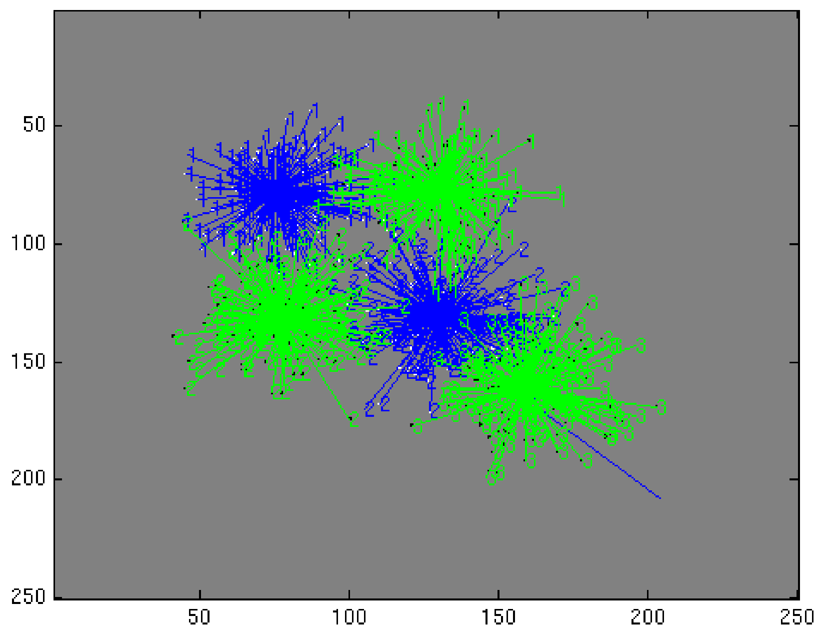

Figure 9: **Intra-class clustering result.** The intra-class clustering results on the data in figure 1 are shown. The white dots are clustered into two groups (blue) and the dark dots are clustered into three groups (green).

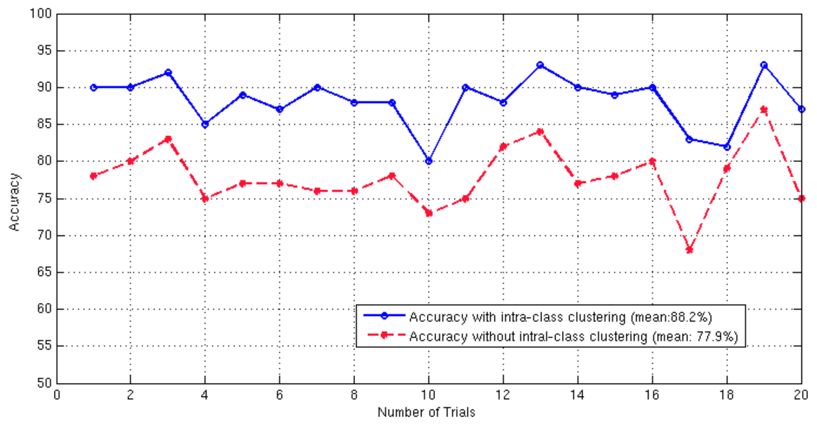

Figure 10: **Enhanced classification performance with intra-class clustering..**

In order to test how much the intra-class clustering can improve the classification accuracy, we compared the results of classification with intra-class clustering to the results without this step. We conducted 20 trials, where within each trial we trained with 1,000 examples and tested with 100. With intra-class clustering, the accuracy is significantly improved. As shown in figure 10, intra-class clustering method wins on every trial, and on average, it has a stable 10% improvement than the trials without this process.

3.3 Experiment 3: Test on real data

To evaluate the efficiency of our algorithm on a realistic task, we tested the intra-class clustering algorithm on datasets obtained from an olfactory database. The data were generated by an olfactory sensor array and contained 60 features. The training set contained 196 records and the testing set contains 45 examples. Figure 11 shows the PCA plot (Jolliff 1986) of the training set along the first and the second principle component dimensions, and figure 12 the LDA plots in the first and second largest component dimensions.

The direct LDA projection as shown in figure 12 does not give a satisfactory result. We can easily observe that classes 1, 2, and 3 are overlapping heavily. As shown in the PCA plot, the non-Gaussian data structure among classes poses a significant difficulty to LDA, so it fails to provide the optimal projection for discriminating the five classes.

On the 2 dimensional space produced by the first and the second largest PCA components, we can apply the intra-class clustering algorithm. This method can separate the overlap in the data and partition each class into several single Gaussian-like distributions. This procedure has promise for improving the LDA performance (figure 13). We tested the classification accuracy by using a quadratic classifier on the 2D LDA-transformed space with intra-class clustering procedure as shown in figure 13. The accuracy was as high as 84.44% on the test dataset. Comparing to quadratic classifier applied directly on 2D PCA-transformed data (figure 11, $accuracy = 33.33\%$) or LDA transform ($accuracy = 77.77\%$), the intra-class

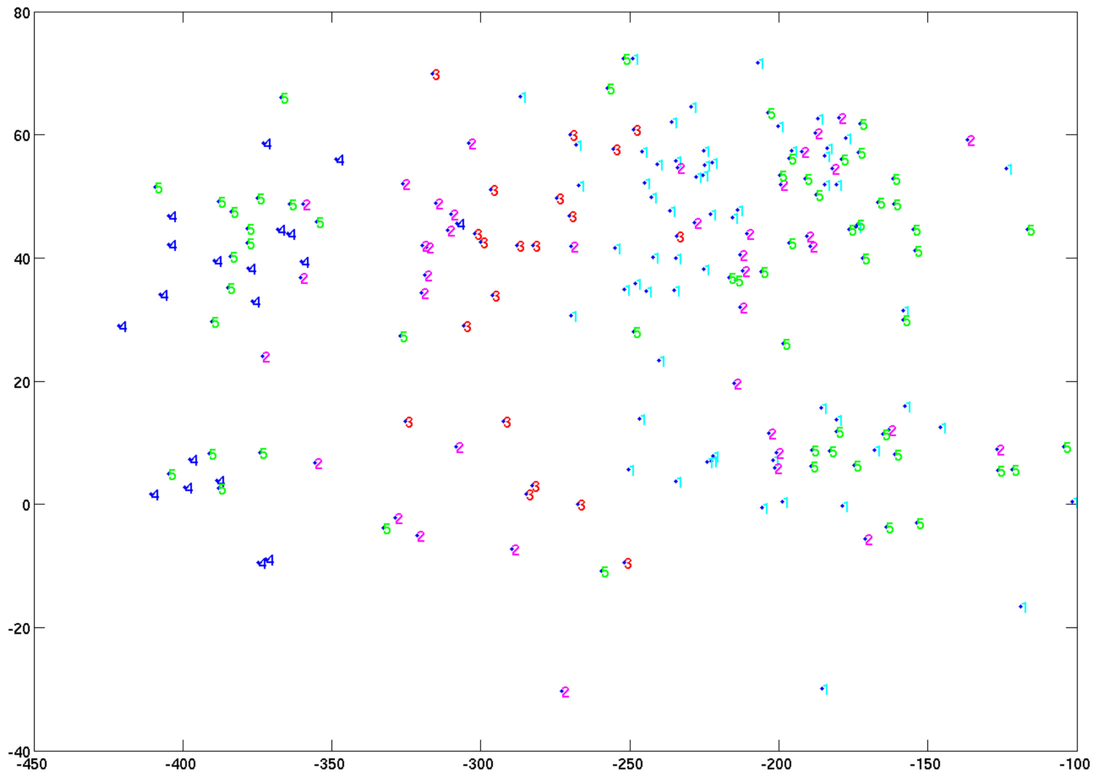

Figure 11: **PCA projections.** Data are labeled by their class numbers.

clustering method showed improved performance.

4 Discussion

Intra-class clustering was first used to improve facial feature detection by (Lucey et al. 2003). In Lucey's approach, a Gaussian mixture models (GMM; Reynolds 1994) was employed to partition the data in one class into several Gaussian distributions. The parameters of each Gaussian model were estimated by the Expectation Maximization (EM) algorithm (Dempster et al. 1977). This approach can effectively separate a non-Gaussian distribution of one class into several Gaussian distributions. However, it does not account for the context information which may be important in drawing a boundary between intra-class clusters.

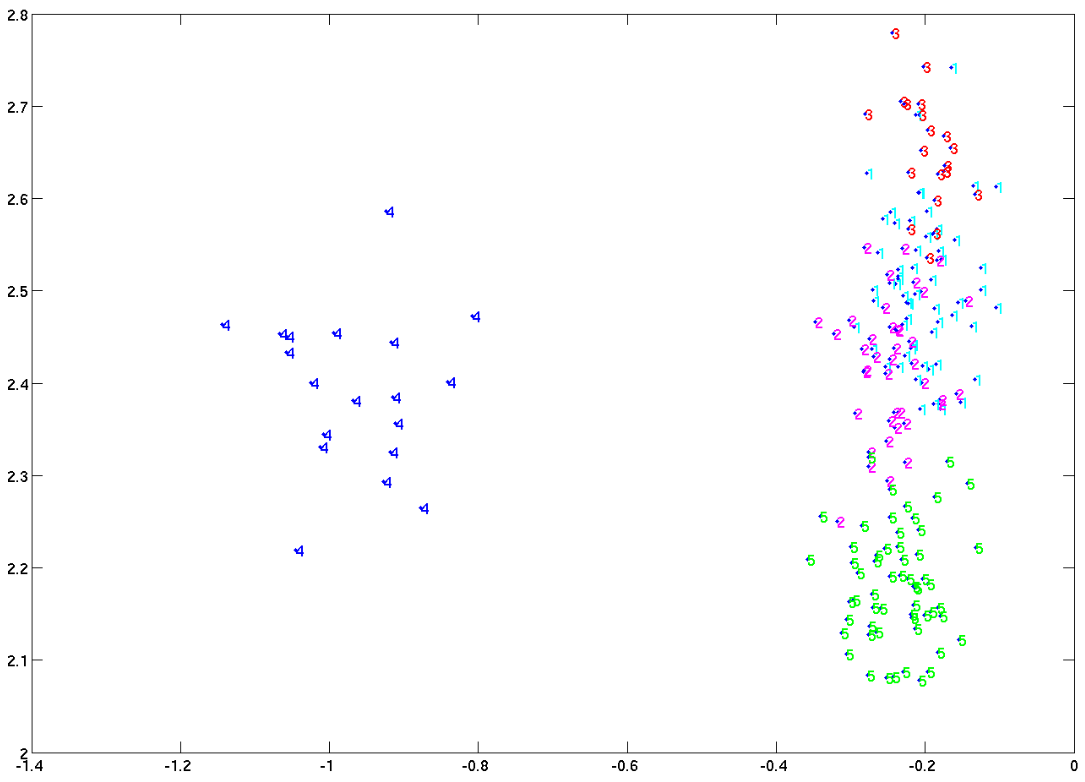

Figure 12: **LDA projections of olfactory data.** Data are labeled by their class numbers. Classes 1, 2 and 3 are overlapping. This view fails to provide the optimal projection for discriminating the five classes.

For example, the problem as illustrated in figures using 2 and 3 cannot be solved by this approach. Moreover, the approach did not aim to separate the means among classes so we cannot not expected it to help LDA completely overcoming its limitations. The intra-class clustering algorithm proposed in this paper is free of the above problems.

Besides the above, our intra-class clustering method is resistant to noise. Using Gaussian as the kernel to estimate the density in the first step can eliminate the noise either in the object class or in the context class. (The density estimation process is equivalent to a convolution process using Gaussian filter, which is a low pass filter.) This design can maximally remove the impact of noise which has a relatively higher spatial frequency.

The algorithm we used to locate the local maximum on the density map is a steepest-

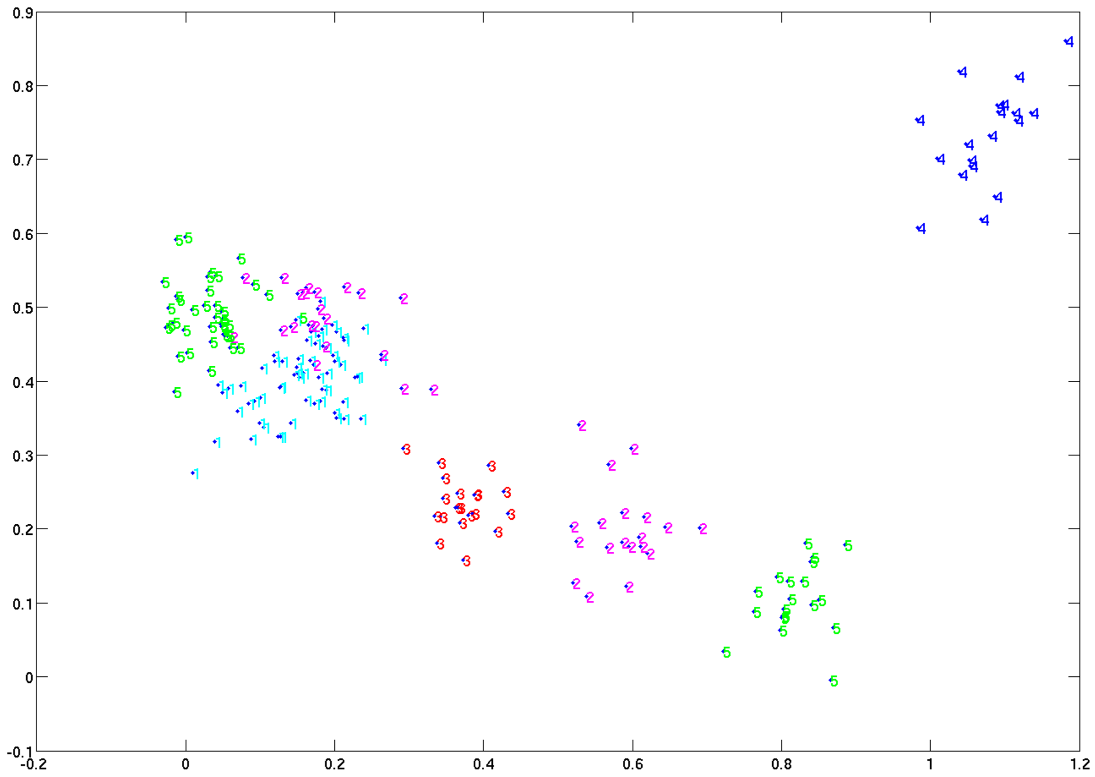

Figure 13: **LDA projection after intra-class clustering procedure.** We use the intra-class clustering to find this LDA plot, and the points are labeled into the original five classes. Now the five classes are well-discriminated and the boundary are clear in this LDA plot. There is no mix up between samples of different classes.

ascent algorithm. Indeed, steepest-ascent hill climbing may suffer from local maxima, ridges, and plateaux in general (Russell and Norvig 2003). Here local maximum is not an issue, because we are in fact specifically looking for it. Ridges and plateaux require infinite points of local maxima in the density map, but when we use uniform Gaussians as kernels to estimate the density, this implies that the examples have to be infinite inside a range. This is untrue in reality, because all the training samples are mapped to discrete points in the sample space, so there are finite examples in any given range. Hence, no plateau or ridge can be formed by discrete samples. Therefore, it is not surprising that we can use a simple steepest-ascent algorithm to locate the associated local maxima and link samples to their intra-class cluster centers located at these local maxima.

The complexity of each steepest-ascent is $O(N)$ (Kowalewski 1995), and this searching process is iterated for each example. One might argue that when applying this intra-class clustering method in high dimensional space, the searching strategy may require tremendous computational resources. We admit that the searching procedure may have such a limitation, but one could (1) reduce the high dimensional searching into lower dimensions using PCA or similar method (usually two or three largest components in PCA give sufficiently good results), or (2) map the continuous space into discrete space and represent it by a matrix in sufficient resolution to preserve necessary structure information. These two methods can significantly reduce the computational demand and make this intra-class clustering algorithm more easily applicable.

5 Conclusion

In this paper, we introduced a novel approach to intra-class clustering. This context-sensitive intra-class clustering method can separate non-Gaussian distributions into multiple clusters of Gaussian-like distributions. At the same time it is context sensitive, thus it can effectively reduce the overlap among resulting subclusters. Hence, each class data may be divided into a number of new subclasses with new class labels based on their spatial structure. The new labeled classes are distributed more Gaussian-like, have less overlap, and have separated means. This procedure has promise in significantly improving the performance of algorithms dependent on Gaussian distribution such as LDA and quadratic classifiers. In sum, this method can be effectively used as a general data preprocessing step for standard supervised learning algorithms.

References

Dempster, A., Laird, N., and Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Royal Statistical Society*, 39:1–38.

Duda, R. O., Hart, P., and Stork, D. G. (2001). *Pattern Classification.* New York: Wiley. Second edition.

Hader, S., and Hamprecht, F. A. (2003). *Efficient density clustering using basin spanning trees.* Springer.

Jolliff, I. (1986). *Principal Component Analysis.* Springer.

Kowalewski, F. (1995). A gradient procedure for determining clusters of relatively high point density. *Pattern Recognition*, 28:1973–1984.

Lucey, S., Sridharan, S., and Chandran, V. (2003). Improved facial feature detection for AVSP via unsupervised clustering and discriminant analysis. *EURASIP Journal on Applied Signal Processing*, 2003:3:264–275.

Paren, E. (1962). On estimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33:1065–1076.

Reynolds, D. (1994). Experimental evaluation of features for robust speaker identification. *IEEE Transactions on Speech and Audio Processing*, 2:539–564.

Russell, S., and Norvig, P. (2003). *Artificial Intelligence A Modern Approach.* Upper Saddle River, New Jersey: Pearson Education. Second edition.

Silverman, B. (1986). *Density Estimation for Statistics and Data Analysis.* Chapman and Hall.

Zhao, W. Y. (2000). Discriminant component analysis for face recognition. In *15th Int. Conf. On Pattern Recognition*, 818–821.