An Adaptive Classification Algorithm Using Robust Incremental Clustering

Herward Prehn and Gerald Sommer
Computer Science Institute, Christian-Albrechts-University of Kiel, Germany
{hp, gs}@ks.informatik.uni-kiel.de

Abstract

In this paper we present an adaptive classification method that features a robust, efficient and simple to use incremental clustering algorithm. A new assignment strategy for incorporating new data patterns allows clusters to align more exhaustively with the data structure. This almost eliminates the sensitivity to the order of input data, many incremental clustering algorithms suffer from, reduces the number of clusters needed and thus improves also time efficiency. For updating the clusters' representations we utilize an incremental version of PCA which generates its learning rate automatically from the number of patterns. Furthermore, the size and number of clusters is controlled by the classification error. So we get a classification method where nothing but the target error needs to be pre-specified. We conducted experiments on artificial and real data to demonstrate the capabilities of the proposed algorithm.

1 Introduction

Clustering and classification problems arise in applications across many scientific disciplines when dealing with large databases. In both cases one is interested in representing the data in an abstract and compressed form that improves organizing and understanding the data. Clustering is the unsupervised process of grouping data patterns that are similar w.r.t. a certain metric or taxonomy. Whereas in classification representations for classes of labeled patterns have to be learned, which in turn are used to label newly encountered patterns [5].

In the COSPAL[1] project we are confronted with the classification of a dynamic database. Within this project we develop a learning computer vision system that creates and constantly modifies a database of appearance-based object features to recognize and classify objects in camera images.

[1]The work presented here was supported by the the European Union, grant COSPAL (IST-2004-71567). However, this paper does not necessarily represent the opinion of the European Community, and the European Community is not responsible for any use which may be made of its contents.

Classifying data patterns can be seen as two problems: first, finding an abstract representation for the patterns of each class and second, keeping these representations disjoint. For the first problem we choose an incremental clustering algorithm. Thus, we benefit from the fact that even non-convex data distributions can be approximated by a set of simple mathematical concepts like e.g. hyperellipsoids. And we take care of the dynamic nature of the database. To solve the second problem we incorporate the classification error to control the growth of clusters.

Concerning incremental clustering algorithms there are four common issues that have to be considered. The first is the selection of a threshold θ called 'vigilance' parameter. It defines the range around the cluster's centroid in which patterns have to ly for being integrated into a cluster. This is comparable to the problem of the user-specified number of clusters in non-incremental methods like the k-means algorithm. The second issue is the sensitivity to the order of input data. Typically, clustering results differ significantly for patterns presented in different sequences. The third issue is the information loss due to the abstraction model chosen to summarize a cluster. Finally, incremental methods need parameters like learning rates or similar to weight the impact of new data samples and to ensure convergence.

In this paper we propose an adaptive classification algorithm including a new incremental clustering method. The later follows a new assignment strategy for generating the clusters making it more robust w.r.t. to the order of input data. This also lets clusters grow more efficiently into the data, so less clusters are needed and time complexity is reduced. Furthermore, the update algorithm does not require a manually set learning rate, but derives it from the number of input patterns. Combined with an automatic selection of the vigilance parameter this makes our method easy to use.

The remaining of this paper is organized as follows. In section 2 previous incremental clustering algorithms are summarized as well as an incremental version of principal component analysis which will be employed. Section 3 describes the proposed classification and clustering algorithms. In section 4 experimental results are presented and section 5 concludes the paper.

2 Previous Work

2.1 Incremental Clustering Algorithms

One of the first incremental clustering algorithms was the Leader algorithm [4]. It uses the aforementioned vigilance parameter θ to determine whether a new pattern can be assigned to an existing cluster or should form a new one by itself. Many algorithms follow this principle for clustering data instances incrementally. The Leader algorithm has gained popularity because of its neural network implementation, the ART network [2]. In a neural network clusters are represented by single neurons, where the neuron's weights represent the mean pattern and the range of the cluster.

In [6] Su and Liu present a further development of the ART network. For a more concise representation of the data they model the input domains of the neurons as hyperellipsoids rather than hyperspheres. For this purpose Su and Liu replace the linear neurons by quadratic ones. Their algorithm operates in two steps. In the first step the network is initialized and the neurons are updated or created by presenting the patterns one by one. In the second step they determine the optimal value for the vigilance parameter. Therefore the neurons' input ranges are increased by changing θ with defined step size. As two neurons start to overlap they are merged into one. It is observed how the number of neurons changes with the size of θ . The optimal value for θ , and thus the optimal cluster size and number, is chosen from the largest interval for which the number of neurons stays constant.

A method that reduces the sensitivity to the order of input data is presented in [3], the GRIN algorithm. Here the patterns are merged into clusters following the law of gravitation. According to a statistics-based test for the quality of a single cluster, clusters are split or merged and then organized in a hierarchical dendrogram. From the dendrogram clustering results can be obtained at various levels of abstraction. In contrast to the method presented in [6], the GRIN algorithm also allows for future re-structuring of the dendrogram, not only at the level of single clusters.

2.2 Incremental Principal Component Analysis

In our proposed clustering algorithm we need to estimate the main directions of variation in the data set of each cluster in an incremental way. Therefore, we employ the candid covariance-free incremental principal component analysis (CCIPCA) proposed by Weng et. al. [7].

Given a data set $X = \{x(1), ..., x(n)\}$, $x(i) \in \mathbb{R}^d$ with zero mean[2], one can compute the first k dominant eigenvectors $v_1(n)$, $v_2(n)$, ..., $v_k(n)$ directly from the patterns $x(n)$, $n = 1, 2$, ... where $v_i(j)$ specifies the estimate for the ith eigenvector after presenting the jth pattern $x(j)$. For updating the eigenvectors from the nth pattern $x(n)$ equations (1) and (2) are evaluated with $i = 1$, ..., k

$$v_i(n) = \frac{n-1}{n} v_i(n-1) + \frac{1}{n} x_i(n) x_i^T(n) \frac{v_i(n-1)}{\|v_i(n-1)\|} \quad (1)$$

$$x_{i+1}(n) = x_i(n) - x_i^T(n) \frac{v_i(n)}{\|v_i(n)\|} \frac{v_i(n)}{\|v_i(n)\|}. \quad (2)$$

Initially, $x_1(n)$ is set to $x(n)$ and for $i = n$, $v_i(n)$ is set to $x_i(n)$. Equation (1) can be interpreted as pulling the eigenvector towards the new pattern by adding a weighted version of the pattern vector to the eigenvector. The weight is composed by the factor $\frac{1}{n}$ and the projection of $x_i(n)$ onto $v_i(n-1)$, which is exactly the impact of the pattern within the direction of $v_i(n-1)$. For the computation of higher order eigenvectors the pattern is reduced by its impact (projection) onto the lower order eigenvectors according to equation (2).

3 The Proposed Algorithm

3.1 Adaptive Classification Algorithm

This section describes the intialization steps of the proposed classification algorithm.

First, the vigilance parameter θ is set to a large value w.r.t. an average distance between two data points. Then, for all data patterns of the same class the clustering algorithm described in section 3.2 is invoked to find the representation of that class. Next, the clusters have to be reorganized according to the method defined in section 3.3 to achieve a more concise data representation. In a fourth step the classification error is evaluated on a testing data set. While the error is greater than a specified target θ is decreased and clustering and re-organizing are repeated for each class. By decreasing θ the range of the individual clusters is narrowed which improves the fitting on the data, but worsens the generalization ability. As in the extreme case of total over-fitting each pattern is modeled by its own cluster, convergence is guaranteed.

Once the target error is reached, the classifier has been initialized. If during the use new labeled patterns arrive, they are integrated by applying the clustering algorithm, so constant learning is realized.

3.2 Incremental Clustering Using Hyperellipsoids

Concerning data abstraction we have chosen to model individual clusters by hyperellipsoids. Their mathematical concept is still sufficiently simple, but they are more flexible than hyperspheres, so that less clusters are needed to

[2]The mean may be incrementally estimated and subtracted out.

reach the same level of approximation for a given set of patterns. This is important as time complexity of incremental clustering algorithms is directly proportional to the number of clusters. The axes of the hyperellipsoids are aligned with the eigenvectors of the patterns forming the cluster and have the magnitude of the corresponding variance scaled with θ.

The actual process of clustering is the following: For a new pattern x the distances $r_j(x)$ to all existing clusters $j = 1, ..., J$ are calculated. Therefore, x is first transformed into the eigenspace of cluster j:

$$\hat{x} = \bar{V}_j^T \left(x - b_j \right), \tag{3}$$

and then the distance, weighted w.r.t. the scaled eigenvectors, is computed

$$r_j(x) = \sum_{i=1}^{d} \left(\frac{\hat{x}_i}{\theta \sqrt{\|v_{j,i}\|}} \right)^2. \tag{4}$$

$\bar{V}_j$ is the matrix containing the unit eigenvectors, b_j the mean and $v_{j,\,i}$ the ith eigenvector of cluster j.

In case that $r_j(x) \leq 1$, cluster j is updated employing the CCIPCA algorithm described in section 2.2. If no clusters exist or $r_j(x) > 1 \ \forall j$, a new cluster is initialized by setting $b_{J+1}(1) = x$ and $v_{J+1,\,1}(1) = x$.

It has to be emphazised that in contrast to the neural network presented in [6] we do not have the difficulty of selecting learning rates for a gradient-based update function. In our approach the learning rate is defined by the ratio of the factors $\frac{n-1}{n}$ and $\frac{1}{n}$ which are automatically set by the number of patterns and ensure convergence as n gets large [7].

3.3 Re-organize Clusters

To achieve robustness w.r.t. the order of input not only the best-matching cluster, but all matching clusters are updated with a new pattern. This allows clusters to grow into the data. In figure 1(a) it can be seen that the development of clusters following the winner-takes-all strategy, as proposed in [4, 6], yields only small clusters that are not related to the distribution of the data. Whereas the clusters in figure 1(b), produced by our algorithm, show a strong correlation to the data. As during their generation many clusters incorporate the same patterns, they more or less cover the same space (the space of the patterns). Once the cluster models are aligned with the data, the redundance is eliminated by deleting clusters whose domain is also covered by other clusters by more than δ_1 percent. The indices of the patterns that loose their coverage are collected for later re-assignment. Analyzing the coverage of the data can be efficiently done employing an $J \times N$ assignment matrix A, where J is the number of clusters and N the number of patterns. The elements a_{ij} are set to zero or one in the case that

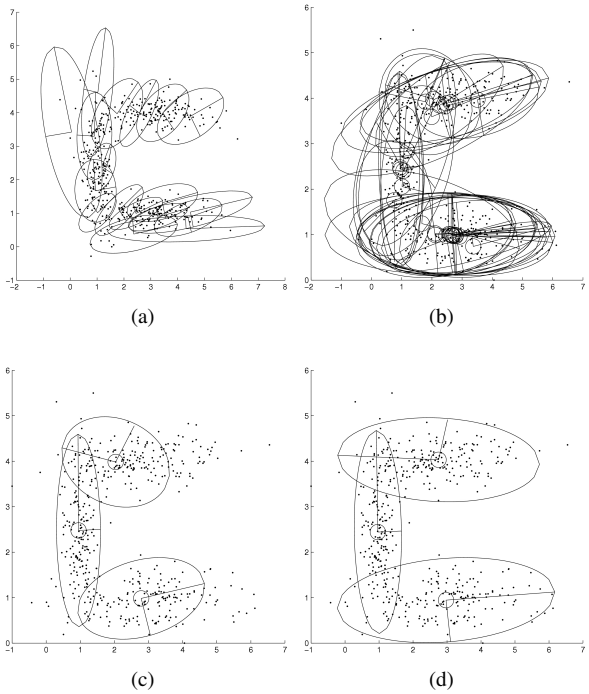

Figure 1. Cluster developement following different assignment strategies: Updating only best (a) or all (b) matching clusters. Results after deleting overlapping clusters (c) and after re-assigning patterns (d).

cluster i covers pattern j. The matrix A is also used to care for outliers by deleting clusters that cover less than δ_2 percent of N. During the re-assignment only the best-matching of the remaining clusters is updated by each pattern. Thus, we avoid creating new ill-shaped clusters. Figures 1(c) and 1(d) show the results of reduction and re-assignment, respectively. To show the robustness of our method w.r.t. the selection of the parameters δ_1 and δ_2, we have set $\delta_1 = 0.5$ and $\delta_2 = 0.01$ for all experiments presented in this paper.

4 Experimental Results

In this section we present the results of two experiments. One on artificial data to demonstrate the algorithms capabilities in principle and one on real data from our application. For both experiments we use half of the data as training set and the other half for testing.

Figure 2 shows the clustering and classification results on three non-convex data classes. It can be seen that the clusters align well with the data, so that only few are needed to approximate the patterns. Thus, classification time is saved, as for each pattern fewer similarity calculations have to be carried out. These results have been achieved by setting the target error to $\varepsilon = 0.01$. Four of the 550 testing patterns have been misclassified (marked with circles).

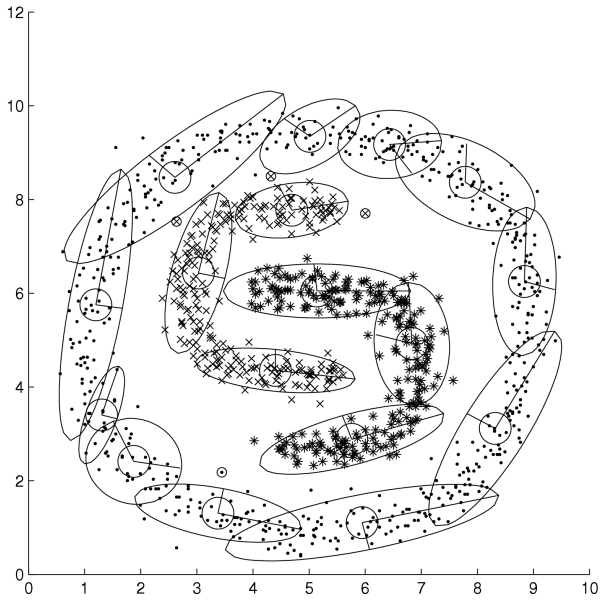

Figure 2. Classification result for non-convex data. Ellipsoids show good matching quality.

The proposed algorithm also yields good abstraction for real data. In figure 3 RGB values are displayed in the 3D RGB cube as they occur in a color segmentation scenario of our computer vision system. With just 11 clusters (centers marked with circles) even on this data an error of less than $\varepsilon = 0.01$ could be reached.

5 Conclusion

In this paper a solution to classification problems for dynamic databases is presented. Its main contributions are w.r.t. finding the abstract representations for the individual pattern classes, time complexity and the simplicity of using the method. The first two aspects are tackled by the new assignment strategy of the proposed incremental clustering algorithm. This strategy allows all clusters in the vicinity of a new pattern to incorporate it. Thus, all clusters get the chance to align their domains with the structure of the data. These fully developed clusters represent the data more concisely than small and ill-shaped ones, resulting from winner-takes-all assignments. As time complexity is proportional to the number of clusters it can be reduced significantly. Furthermore, the sensitivity to the order of the input data, an inherent problem of incremental algorithms, is reduced, as all clusters get the chance for full development. The simple use of the method is achieved by employing the CCIPCA to update the clusters as it is free of parameters that need to be tuned. We have shown that the algorithm performs well on artificial and real data of different dimensionalities.

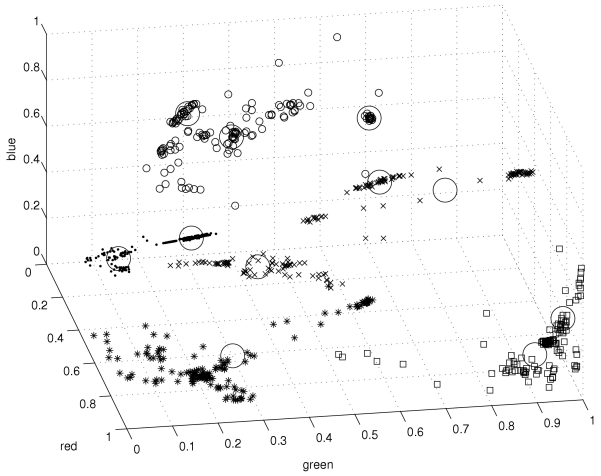

Figure 3. Five classes of RGB values are modeled with 11 clusters (only cnters shown) achieving a classification error of $\varepsilon < 0.01$.

Despite all positive, there is still room for improvements. So far all patterns have to be kept in memory for the purpose of re-organizing the clusters. We believe that this can be avoided and re-organizing can be done solely on the abstract summary of the clusters. Furthermore, the introduction of a quality measure for clusters can improve the process of deleting overlapped clusters. We also want to investigate which profits can be made by combining our approach with the dynamic cell structure (DCS) proposed in [1]. Besides similar advantages, the DCS also includes neighbourhood relations which could yield further improvements.

References

[1] J. Bruske and G. Sommer. Dynamic cell structure learns perfectly topology preserving map. *Neural Computation*, 7(4):845–865, 1995.

[2] G. A. Carpenter, S. Grossberg, and D. B. Rosen. Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks*, 4:759–771, 1991.

[3] C. Y. Chen, S. C. Hwang, and Y. J. Oyang. A statistics-based approach to control the quality of subclusters in incremental gravitational clustering. *Pattern Recognition*, 38(12):2256–2269, Dec. 2005.

[4] J. A. Hartigan. *Clustering Algorithms*. John Wiley and Sons, New York, 1975.

[5] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Comput. Surv.*, 31(3):264–323, 1999.

[6] M. C. Su and Y. C. Liu. A new approach to clustering data with arbitrary shapes. *Pattern Recognition*, 38(11):1887–1901, Nov. 2005.

[7] J. Weng, Y. Zhang, and W.-S. Hwang. Candid covariance-free incremental principal component analysis. *IEEE Trans. Pattern Anal. Mach. Intell*, 25(8):1034–1040, 2003.