Selection of prototype rules: context searching via clustering

Marcin Blachnik[1] Włodzisław Duch[2] and Tadeusz Wieczorek[1]

[1] Division of Computer Methods, Department of Electrotechnology and Metallurgy, The Silesian University of Technology, Krasińskiego 8, 40-019 Katowice, Poland
[2] Department of Informatics, Nicolaus Copernicus University, Grudziądzka 5, Toruń, Poland, and School of Computer Engineering, Nanyang Technological University, Singapore.
WWW home page: Google "Duch"

Abstract. Prototype-based rules are an interesting alternative to fuzzy and crisp logical rules, in many cases providing simpler, more accurate and more comprehensible description of the data. Such rules may be directly converted to fuzzy rules. A new algorithm for generation of prototype-based rules is introduced and a comparison with results obtained by neurofuzzy systems on a number of datasets provided.

1 Introduction

Similarity-based approaches that developed from the k-Nearest Neighbors (kNN) algorithm [1, 2] are still one of the most useful and popular algorithms in pattern recognition. They are also at the foundation of the Artificial Intelligence learning methods, such as the Case Based Reasoning or Memory Based Reasoning methods, allowing for comparison of complex objects that cannot be easily represented in a feature space with fixed number of dimensions.

Nearest neighbor algorithms, or more generally Similarity-Based Learning (SBL) framework [2, 3], may provide not only predictive models, but also prototype-based logical rules (P-rules) [4, 5] for data understanding. Knowledge hidden in many datasets can be captured in a small set of prototypes using appropriate similarity measures. As shown in [6] this type of rules are equivalent to fuzzy rules (F-rules). P-rules seem to be more general because they support all types of features (discrete, nominal or symbolic), while the use of F-rules is restricted to continuous or ordinal attributes. Moreover, non-additive distance functions may include explicit models of correlation between variables and cannot be converted into fuzzy rules. Algorithms for selection of good prototypes and similarity measures are in a direct competition to the popular neurofuzzy approaches and fuzzy modeling techniques [7], and therefore their further development is an important issue.

Selection of reference vectors, also called prototypes, is very important especially for large datasets, because storing the whole training set in memory is then prohibitively expensive and searching for nearest neighbors becomes computationally very expensive, making similarity-based algorithms completely useless for real time applications. The need to overcome these limitations brought the development of many approaches

aimed at increase of the speed and reduction of the memory requirement of the basic nearest neighbor algorithms. As a result many powerful algorithms for prototype selection were created [8–11] that may also find an application in creation of P-rules.

These algorithms may be divided into four groups: noise filters, data condensing methods, clustering, and prototype optimization methods. Most of these approaches try to reduce the number of prototypes by getting rid of irrelevant training vectors [10]. Some algorithms try also to identify and remove outliers that decrease accuracy. More advanced algorithms like WITS, DROP [1–5, 11], remove also vectors which lie close to the decision border and are not necessary for classification, increasing generalization abilities. Another group of methods starts with a small number of prototypes optimizing their position using such techniques as learning vector quantization (LVQ). This is a powerful method, but the results strongly depend on the starting position of prototypes. Results presented in [8, 9] lead to the conclusion that LVQ approach should be used as second step following other prototype selection methods.

The last group of methods is based on clustering algorithms that search for interesting groupings of vectors in a dataset [1]. Usually clustering is done separately for each class, so that each class has an independent number of clusters. In this approach spherical clusters are preferred like obtained from Hard C-means (HCM), Fuzzy C-Means (FCM) algorithm or Gaussian Mixture Model (GMM). Unfortunately these clustering methods search for clusters among all vectors belonging to one of the classes, producing a subset of irrelevant prototypes that are often far away from the decision boundary and that do not participate in the decision process.

This problem is especially important for P-rules where a small subset of reference vectors that can be interpreted as prototypes is searched for [2]. These observations allow for construction of a new prototype selection algorithm based on the context dependent clustering approach, for example the Conditional FCM (CFCM) with weighting of the training vectors [12]. In the next section heterogeneous distance functions are described, in the third section CFCM algorithm is presented, in the fourth section creation of an appropriate condition values to determine clustering context is described, the algorithm for the selection of optimal number of prototypes is presented in section five, followed by empirical experiments in section six, and the final section contains conclusions.

2 Heterogeneous distance functions

Real word classification problems usually involve different types of features, some may be continuous or linearly ordered, some may be discrete and some symbolic. This constitutes a real problem for many algorithms, including statistical, neural or neurofuzzy approaches (ANFIS). Similarity-Based Learning framework [2, 3] defines many variants of similarity evaluations, including different distance functions for different types of attributes. The most popular and frequently used Minkovsky distance function is a generalization of Euclidian distance and is applicable only for numerical attributes. Symbolic features require different distance functions that may be defined using conditional probabilities to evaluate similarity of each symbolic feature value from the point of view of class distributions, as it is done in the Value Distance Metric (VDM) [15].

Numerical and probabilistic distance measures may be combined in a general distance metric, usually called the Heterogeneous Distance Metric (HDM), and in particular case of the VDM metric the Heterogeneous Value Distance Metric (HVDM). Such distance functions were used by Wilson and Martinez [16] who simply added scaled contributions from different features:

$$D\left(\mathbf{x},\mathbf{r}\right)^{\alpha}=\sum_{i=1}^{n}d\left(x_i,r_i\right)^{\alpha} \tag{1}$$

where $D\left(\mathbf{x},\mathbf{r}\right)$ is total distance between two vectors **x** and **r** with n features, $d(x_i,r_i)$ are distances calculated for single features, and α is an exponent (in more general case exponents on the left and right side of Eq. 1 may be different [3]). Explicit account of correlations between features is done in the simplest way by introducing covariance matrices. HDM assumes that different types of features are independent, and for the real-valued or ordered discrete features Minkovsky's distances are used, while for symbolic features probabilistic distances based on conditional probabilities are used.

$$D_{Mink}\left(\mathbf{x},\mathbf{r}\right)^{\alpha}=\sum_{i=1}^{n_1}\left|x_i-r_i\right|^{\alpha} \tag{2}$$

$$D_{VMD}\left(\mathbf{x},\mathbf{r}\right)^{\alpha}=\sum_{i=1}^{n_2}\sum_{j=1}^{C}\left|p\left(c_j|x_i\right)-p\left(c_j|r_i\right)\right|^{\alpha} \tag{3}$$

where n_1 is the number of numerical and n_2 of symbolic values, C is the number of classes and posterior probabilities $p\left(c_j|x_i\right)$ and $p\left(c_j|r_i\right)$ are calculated as:

$$p\left(c_j|x_i\right)=Nx_{ij}/Nx_i \tag{4}$$

where Nx_i is number of instances in the training set that have value x for feature i, and Nx_{ij} is the number of training vectors from class j that have value x for feature i.

Because HVDM is additive it can be written as the sum of two distance functions that depend on different attribute types:

$$D\left(\mathbf{x},\mathbf{r}\right)^{\alpha}=D_{Mink}\left(\mathbf{x_a},\mathbf{r_a}\right)^{\alpha}+D_{VDM}\left(\mathbf{x_b},\mathbf{r_b}\right)^{\alpha} \tag{5}$$

where $\mathbf{x}_a$ and $\mathbf{r}_a$ are subsets of numerical attributes and $\mathbf{x}_b$ and $\mathbf{r}_b$ are subsets of their symbolic features. An important problem with such heterogeneous functions is to define a common distance scale. D_{Mink} takes values in the range $(0,+\infty)$ while D_{VDM} in the range $\left(0,n_2^{1/\alpha}\right)$. Features should be properly normalized to assure that each distance component has the some or comparable contribution. Wilson and Martinez [16] proposed one type of normalization for numerical attributes (6) and three different normalization methods for VDM distance (7), where the choice depends on particular problem:

$$d_{Mink}\left(x,r\right)=\frac{\left|x-r\right|}{4\sigma} \tag{6}$$

$$
\begin{aligned}
N1 : d_{VMD}(x,r) &= \sum_{j=1}^{C} \left| \frac{Nx_j}{Nx} - \frac{Nr_j}{Nr} \right| \\
N2 : d_{VMD}(x,r) &= \sqrt{\sum_{j=1}^{C} \left| \frac{Nx_j}{Nx} - \frac{Nr_j}{Nr} \right|^2} \\
N3 : d_{VMD}(x,r) &= \sqrt{C \cdot \sum_{j=1}^{C} \left| \frac{Nx_j}{Nx} - \frac{Nr_j}{Nr} \right|^2}
\end{aligned}
\tag{7}
$$

In $N1$ the distance between one dimensional probabilities is calculated using the Manhattan norm, while in $N2$ and $N3$ the Euclidean norm is used.

In [16] authors suggest that normalization $N2$ "favors having all of the class correlations fairly similar rather then having some very close and some very different". The difference between $N2$ and $N3$ normalizations may have an influence on classification using kNN method, but for P-rules, with a small number of reference vectors common feature weighting may be replaced by optimization of their positions. The parameter α that occurs in all distance functions has significant influence on the shape of decision borders [3]. There is no specific value of this parameter that will always lead to the best classification results, therefore the optimal α value should be determined using meta-learning algorithms [17]. Another possibility is to set the value of α depending on particular needs. This is one of the P-rules advantages, because the results are equivalent to fuzzy rules with various types of membership functions, for example Gaussian for α=2 or crisp logical rules for α=∞.

3 Conditional Fuzzy C-Means

In classification tasks the area between samples from different classes is most important [1], and the goal of pattern recognition algorithms is to construct the optimal shape of this border to assure maximum generalization, that is the lowest error rate for the test data. This observation encourages searching for prototypes similar to "support vectors" [18] in the border area. Classical clustering algorithms search for groups of data in the whole data space without any knowledge of class distribution. Of course one may cluster each class independently, however the centers of the clusters will lead to prototypes far from the decision borders. Another disadvantage of classical clustering algorithms follows from the fact that useless prototypes are created for data clusters far from decision borders, as show in Fig. 1. Optimization methods from the LVQ family cannot move such prototypes to areas where they could take an important part in the decision process.

One possible solution of this problem is based on conditional or context dependent clustering algorithms. One of the examples of such methods is the Conditional Fuzzy C-Means method (CFCM) [12]. It is an extension of the FCM clustering algorithm where data are grouped under some external conditions defined for every training vector. These conditions are specified by an external variable $y_k$which corresponds to each training vector x_k. This variable y_k has an associated membership function $\mu(y)$ or in other words weight that creates clustering condition f_k=$\mu_A(y)$ $\in$[0,1] defined for every

vector x_k. This condition allows for clustering related data, where f_k defines strength of the relation between data vectors x_k and the external variable y_k.

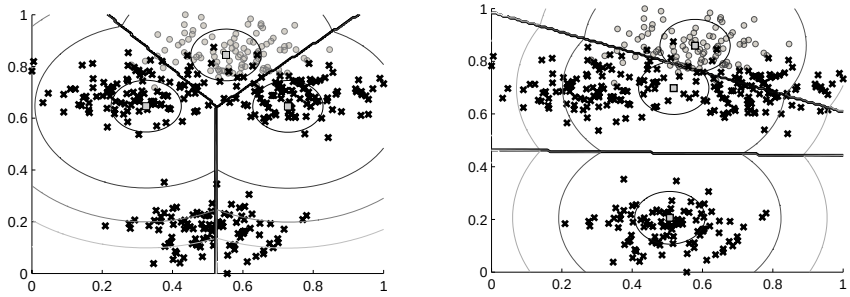

Fig. 1. Decision borders and prototype activation areas for two class problem with prototypes selected by a) Conditional Fuzzy C-means clustering algorithm and b) Fuzzy C-means method.

In our approach this external variable is defined as a measure of distance between each data vector and a possible decision border. The task is to find clusters near to this border. FCM and CFCM are based on minimization of a cost function defined as:

$$J_m(\mathbf{U},\mathbf{V}) = \sum_{i=1}^{C}\sum_{k=1}^{N} (u_{ik})^m \left\|x_k - v_i\right\|_A^2 \tag{8}$$

where C is the number of clusters centered at v_i, N is the number of vectors, m >1 is a parameter, and **U**=(u_{ik}) is a $C \times N$ dimensional membership matrix with elements $u_{ik} \in$ [0,1] defining the degree of membership of the k-th vector in the i-th class. The matrix **U** has to fulfill three conditions:

1^o each vector x_k belongs to the i-th cluster to a degree between 0 to 1:

$$\bigvee_{1\le i\le C}\ \bigvee_{1\le k\le N} u_{ik} \in [0,1] \tag{9}$$

2^o sum of the membership values of k-th vector x_k in all clusters is equal to f_k

$$\bigvee_{1\le k\le N} \sum_{i=1}^{C} u_{ik} = f_k \tag{10}$$

3^o no clusters are empty.

$$\bigvee_{1\le i\le C} 0 < \sum_{k=1}^{N} u_{ik} < N \tag{11}$$

Cost function (8) is minimized under these conditions by [12]:

$$\underset{1\le i\le C}{\forall} v_i = \sum_{k=1}^{N} (u_{ik})^m x_k \Big/ \sum_{k=1}^{N} (u_{ik})^m \tag{12}$$

$$\underset{\substack{1\le i\le C\\ 1\le k\le N}}{\forall} u_{ik} = f_k \Big/ \sum_{j=1}^{C} \left(\frac{\|x_k - \nu_i\|}{\|x_k - \nu_j\|} \right)^{2/(m-1)} \tag{13}$$

4 Determining the context

Searching for optimal position of prototypes for P-rules is a difficult problem; moreover, a balance between the number of prototypes (simplicity of the rules) and the accuracy of the whole system is very important. Irrelevant prototypes, that is prototypes that do not have any influence on classification, should be removed. This assumption allows for search of prototypes that lie close to one of the opposite classes, that is close to the possible decision border.

To determine position of prototypes using conditional clustering algorithm a coefficient w_k is defined, evaluating for a given vector x_k the ratio of a scatter between this vector and all vectors from the same class, divided by a scatter for all vectors from the remaining classes:

$$w_k = \sum_{j,\omega(x_j)=\omega(x_k)} \|x_k - x_j\|^2 \Big/ \sum_{l,\omega(x_l)\neq\omega(x_k)} \|x_k - x_l\|^2 \tag{14}$$

Here $\omega(x_k)$ denotes class label of the vector x_k. This coefficients is then normalized to be in the range [0,1]:

$$w_k = \Big(w_k - \min_k (w_k) \Big) \Big/ \Big(\max_k (w_k) - \min_k (w_k) \Big) \tag{15}$$

Normalized w_k coefficients reach values close to 0 for vectors inside large homogenous clusters, and close to 1 if the vector x_k is near the vectors of the opposite classes and far from other vectors from the same class (for example if it is an outlier). To avoid effects related to multiple density peaks in non-homogenous data distributions a cut-off point for distance of the order of standard deviation of the whole data could be introduced, although in our tests it was not necessary. These normalized weights determine the external variable which then is filtered to assign appropriate context or condition for CFCM clustering. Filtering can be also understood as assigning appropriate linguistic term in the fuzzy sense. In the algorithm used below a Gaussian filter was used:

$$f_k = \exp\left(\frac{w_k - \mu}{\sigma^2} \right) \tag{16}$$

with the best parameters in the range of μ=0.6-0.8 and σ=0.6-0.9, determined empirically for a wide range of datasets. The μ parameter controls where the prototypes will be placed; for small μ they are closer to the center of the cluster and for larger μ closer to the decision borders. The range in which they are sought is determined by the σ parameter.

5 Prototype racing algorithm for optimizing number of prototypes

Most algorithms for clustering require specification of the number of clusters. This is especially important for the C-means clustering group of algorithms, where premises for determining C do not exist, while in the hierarchical clustering they can be obtained by analyzing dendrogram tree [19]. In our approach clusterization of data from each class is done in an independent way, therefore determination of the number of prototypes requires control of several C_i parameters, one for each class. Moreover, these parameters may be strongly correlated. The simplest way to build a model with optimal number of reference vectors is to check its performance for all possible combinations of C_i with optimization of final positions of all prototypes but this will lead to a very high computational costs.

One possible solution is provided by the racing algorithm [13] based on the results of candidate model comparison in cross-validation task. To speed up the learning process Hoeffding or Bayesian estimation of error bounds of analyzed models is used, rejecting those that are significantly worse than the current best model during the optimization procedure. Another possible improvement leading to reduction of computational costs, especially in optimizing balanced accuracy (BER), is based on the heuristics for increasing the number of prototypes only for the class that has worst accuracy. Adding new prototype to the class with lowest accuracy increases the accuracy for that class but may also reduce overall balanced accuracy. To avoid such problems the racing algorithm keeps only the best models, growing the number of prototypes per class as long as the gain in balanced accuracy justifies the added complexity. This approach significantly increases optimization speed but unfortunately not always finds optimal combination of prototypes per class.

6 Experiments and results

In numerical experiments presented below prototype selection scheme described above was compared with results obtained from classical clustering algorithm without external conditions. All results are from the 10-fold crossvalidation tests with the racing algorithm for determining the number of prototypes. Maximum number of iterations was limited to 15 for both FCM and CFCM clustering. Positions of prototypes obtained from clustering were then optimized with the LVQ1 algorithm [19]. Test results summarized in Tab. 1 were obtained for six benchmark datasets from UCI repository [7], with different number of classes and types of features: *Cleveland Heart Disease*, *Glass*, *Ionosphere*, *Pima Indians Diabetes*, and the *Iris* dataset with all 4 and with two most important features. A small medical dataset *Appendicitis* was also used, with two most

Table 1. Classification results for 6 datasets: balanced accuracy, standard deviation, the number of premises and rules.

Dataset	Decision Tree			CFCM			FCM			NefClass		
	bacc	std	Prem/ C-rule	bacc	std	Prem/ P-rule	bacc	std	P-rule	bacc	std	Prem/ F-rule
Appendicitis	76.0	19.0	2/1	81.1	13.0	2/3	79.7	19.7	2	81.9	12.3	1/4
Heart	78.9	7.7	3/5	76.3 78.9	3.4 5.9	13/4 3/2	75.4	6.8	5	76.6	6.6	1/2
Glass	62.4	9.1	6/8	71.2 70.6	11,6 11.1	9/16 6/14	68.6	10.0	16	64.8	9.4	3/13
Ionosphere	88.7	4.8	2/2	90,1 85.9	5.3 6.8	33/7 2/6	84.8	6.5	6	81.8	9.8	2/4
Diabetes	69.9	4.4	2/2	74.9 73.9	3.1 5.9	8/7 2/3	74.1	4.7	7	71.1	4.9	1/2
Iris	93.3	7.0	2/2	96.0 97.3	4.7 4.7	4/5 2/5	95.3	5.5	5	93.3	4.4	2/3

relevant features selected using SSV decision tree (description of these datasets may be found in [4]).

Crisp rules generated by decision tree using Gini index were used to generate C-rules as a reference for accuracy and complexity comparison. Features selected by the tree were used with the CFCM approach. Results are also compared with the state-of-the-art NefClass [7] neurofuzzy system that generates F-rules. It is based on greedy partitioning for initialization of fuzzy sets. In NefClass the number of fuzzy sets and the maximum number of rules are selected manually. Several available shapes of fuzzy membership functions have been used with different number of fuzzy sets and rules, and the best balanced accuracy results are reported in Tab.1. Crossvalidation results are used to select the model and to estimate expected accuracy, and this model is then trained on all data to obtain the number of rules and premises.

Appendicitis dataset is quite small (21+85 samples) and thus standard deviation for balanced accuracy is very large, therefore the differences are not significant. All solutions are quite simple, including the C-rules from the tree: If F7$<$7520.5 and F4$<$12 then Class 2, else Class 1. For the remaining five datasets best balanced accuracy was achieved using the CFCM algorithm, although for such small datasets standard deviations are quite large and thus even 8% differences (Ionosphere) between P-rules and F-rules are not statistically significant. P-rules generated in an automatic way using the CFCM approach have comparable complexity to the manually optimized F-rules. P-rules may be directly converted to the F-rules [6] and therefore algorithms for their generation offer a competitive approach to the neurofuzzy algorithms.

7 Conclusions

In many data mining applications understanding and transparency of the results are of great importance. Rule-based systems should be flexible to represent data accurately,

and should be easy to understand, implying a small number of accurate rules. Although both P-rules and F-rules may fulfill these conditions for some reason P-rules are not so popular, mistakenly believed to be difficult to understand. On the other hand complex sets of fuzzy rules that no-one even tries to understand are hailed as comprehensible just because they are logical rules. Comparison with neurofuzzy systems shows that P-rule algorithms are frequently capable of generating simpler and more accurate description of data in terms of prototypes. The balance between transparency (rule simplicity and their number) and accuracy should be determined in each task individually, depending on the application.

In this paper a new approach based on the conditional fuzzy C-means approach with determination of the distance to the decision border has been introduced and used for selecting good prototypes, optimizing their position and number. This algorithm has several advantages: it limits the search area to the most probable space, facilitates searching for good prototypes, reduces influence of outliers, reduces the number of irrelevant prototypes, creating only prototypes that have important meaning for classification, and it automatically determines optimal number of prototypes for each class.

Although this approach introduces two new parameters that should be tuned experiments showed that it is not sensitive to their values and leaving these values in the μ=0.6-0.8 and σ=0.6-0.9 range is sufficient. The results presented in Tab.1 show that the proposed coefficient for context clustering may significantly improve the prototype selection based on clustering. For some datasets like Glass or Ionosphere the classification quality increased by almost 20% comparing to the normal clustering, while for the other datasets the increase was more modest.

So far the context dependent clustering algorithm was used only with the CFCM clustering, but applications to other algorithms look also very promising and will be investigated in our future work. Combination of this approach with feature selection based on Relief index should lead to a P-rule system that should easily compete with the best neurofuzzy systems.

Acknowledgement: WD is grateful for the support by the Polish Committee for Scientific Research, research grant 2005-2007.

References

1. R.O. Duda, P.E. Hart, D.G. Stork, *Pattern Classification*, New York: John Wiley & Sons, 2nd ed, 2001.
2. W. Duch, “Similarity based methods: a general framework for classification, approximation and association”. *Control and Cybernetics*, vol. 29(4), pp. 937-968, 2000.
3. W. Duch, R. Adamczak, G.H.F. Diercksen, “Classification, Association and Pattern Completion using Neural Similarity Based Methods”. Applied Mathematics and Computer Science vol. 10(4), pp. 101-120, 2000.
4. W. Duch, R. Setiono, J.M. Zurada, “Computational intelligence methods for understanding of data”. Proc. of the IEEE, vol. 92(5), pp. 771- 805, 2004.
5. W. Duch, K. Grudziński “Prototype based rules - a new way to understand the data”. Proc. of IJCNN 2001, Washington D.C., USA, pp. 1858-1863.
6. W. Duch M. Blachnik “Fuzzy rule-based system derived from similarity to prototypes”, Lecture Notes in Computer Science, vol. 3316, pp. 912-917, 2004.

7. D. Nauck, F. Klawonn, R. Kruse. Foundations on Neuro-Fuzzy Systems. Wiley, Chichester, 1997.
8. N. Jankowski, M. Grochowski, “Comparison of Instance Selection Algorithms I. Algorithms Survey”, Lecture Notes in Artificial Intelligence, vol. 3070, pp. 598-603, 2004.
9. M. Grochowski, N. Jankowski, “Comparison of Instance Selection Algorithms II. Results and Comments”, Lecture Notes in Artificial Intelligence, vol. 3070, pp. 580-585, 2004.
10. J.C. Bezdek, L.I. Kuncheva, “Nearest prototype classifier designs: An experimental study”, *International Journal of Intelligent Systems*, vol. 16(12), pp. 1445-1473, 2001.
11. D.R. Wilson, T.R. Martinez, *Reduction techniques for instance-based learning algorithms*. Machine Learning vol. 38, pp. 257-268, 2000.
12. W. Pedrycz, “Conditional Fuzzy C-Means”, Pattern Recognition Letters, vol. 17, 625-632, 1996.
13. O. Maron, A. Moore, The Racing Algorithm: Model Selection for Lazy Learners. Artificial Intelligence Review, vol. 11, 193-225, 1997.
14. C.J. Mertz and P.M. Murphy, UCI repository of machine learning databases, www.ics.uci.edu/pub/machine-learning-databases.
15. C. Stanfill. D. Waltz, Toward memory-based reasoning. Communications of the ACM, 29(12), 1986, pp 1213-1228
16. D. R. Wilson, T.R. Martinez, Improved Heterogeneous Distance Functions, Journal of Artificial Intelligence Research 6, 1997
17. Duch W, Grudziński K, Meta-learning via search combined with parameter optimization. Intelligent Information Systems, Advances in Soft Computing, Physica Verlag (Springer) 2002, pp. 13-22
18. N. Cristianini and J. Shawe-Taylor, An introduction to support vector machines (and other kernel-based learning methods). Cambridge University Press, 2000.
19. S. Theodoridis and K. Koutroumbas, Pattern Recognition. Academic Press, 3rd Ed, 2006.