Mixed Data Object Selection Based on Clustering and Border Objects

J. Arturo Olvera-López, J. Francisco Martínez-Trinidad, and J. Ariel Carrasco-Ochoa

Computer Science Department
National Institute of Astrophysics, Optics and Electronics
Luis Enrique Erro No. 1, Sta. María Tonantzintla, Puebla, CP: 72840, Mexico
{aolvera,fmartine,ariel}@ccc.inaoep.mx

Abstract. In supervised classification, the object selection or instance selection is an important task, mainly for instance-based classifiers since through this process the time in training and classification stages could be reduced. In this work, we propose a new mixed data object selection method based on clustering and border objects. We carried out an experimental comparison between our method and other object selection methods using some mixed data classifiers.

Keywords: Supervised Classifiers, Object Selection, Clustering, Mixed Data.

1 Introduction

The supervised classification is a process that assigns a class or label to new objects according to their features using a set of previously assessed objects called training set, denoted in this paper as T.

In practice, T contains objects with useless information for the classification task, that is, superfluous objects. Due to the superfluous objects in a training set, it is necessary to select those objects (in T) that give relevant information for the classifier. This selection process is known as object selection. The main goal of an object selection method is to obtain a set $S \subset T$ such that S preserves the classification accuracy.

Several methods have been proposed for solving the object selection problem, the *Condensed Nearest Neighbor* (*CNN*) [1] and the *Edited Nearest Neighbor* (*ENN*) [2] are two of the first proposed methods for object selection. The *CNN* method starts with $S = \varnothing$ and its initial step consists in randomly including in S one object belonging to each class. After the initial step, each object in T is classified (with *k-NN*) using S as training set, if an object O is misclassified then O is included in S to ensure that new objects near to O will be classified correctly. The *ENN* rule consists in discarding from T those objects that do not belong to their k nearest neighbors' class. This method is used as noise filter because it deletes noisy objects, that is, objects with a different class in a neighborhood. An extension of *ENN* is *REEN* (*Repeated ENN*) [3] which applies *ENN* repeatedly until all objects in S have the same class that the majority of their k nearest neighbors.

L. Rueda, D. Mery, and J. Kittler (Eds.): CIARP 2007, LNCS 4756, pp. 674–683, 2007.

Other object selection methods are the *DROP* (*Decremental Reduction Optimization Procedure*) which were proposed in [4], their selection criterion is based on the concept of *associate*. The *associates* of an object O are those objects such that O is one of their k nearest neighbors. These methods discard the object O if its associates can be classified correctly without O.

In [5] the *Iterative Case Filtering algorithm* (*ICF*) was proposed, this method is based on the $Reachable(O)$ and $Coverage(O)$ sets which are the neighborhood set and the associates set described above. *ICF* discards an object O if $|Reachable(O)| > |Coverage(O)|$.

Clustering can be used for object selection [6, 7] so that after splitting T in n clusters, S is the set of centers of each cluster. In [8] the *CLU* object selection method is based on this rule and it was applied to the signature recognition problem.

In a training set, the border objects of a class are located in a region where there are objects from different classes. These objects give useful information to the classifier for preserving the class discrimination regions [4, 5]. On the other hand, interior objects of a class (objects that are not border) could be less useful. In this paper, we propose a mixed data object selection method based on clustering; our method finds and retains border objects and some interior objects.

In order to show the performance of the proposed method, we present an experimental comparison between our method and some other object selection methods using the obtained object sets as training for different mixed data classifiers.

This paper is structured as follows: in section 2 our object selection method is introduced, in section 3 we report experimental results obtained by our method, and finally, in section 4 some conclusions and future work are given.

2 Proposed Method

In a training set, interior objects could be deleted without loosing classification accuracy. In this paper we propose a method called *MOSC* (*Mixed data Object Selection by Clustering*) which finds and retains border objects and some interior objects. The selection criterion in *MOSC* is based on clustering, mainly on non homogeneous clusters.

An homogeneous cluster is a set of objects such that all objects belong to the same class whereas in a non homogeneous cluster there are objects belonging to different classes.

In order to find border objects, the *MOSC* method generates clusters and analyses non homogeneous clusters since border objects are located in regions which contain similar objects belonging to different classes.

In order to handling mixed data, *MOSC* uses the *k-means with similarity functions* algorithm (*kMSF*) [9] for creating clusters. This algorithm is based on the same idea as *k-means* but for comparing objects it uses a similarity function and instead of computing means, it computes representative objects for each cluster. The *kMSF* algorithm determines the representative object in a cluster A_j using the next expression (for more details see [9]):

$$r_{\substack{A_i \\ O_j \in A_i \\ A_q \neq A_i}}(O_j) = \frac{\beta_{A_i}(O_j)}{\alpha_{A_i}(O_j) + (1 - \beta_{A_i}(O_j))} + \eta_{A_q}(O_j) \tag{1}$$

Where

$$\beta_{A_i}(O_j) = \frac{1}{|A_i| - 1} \sum_{\substack{O_j, O_q \in A_i \\ O_j \neq O_q}} \Gamma(O_j, O_q) \tag{2}$$

$$\alpha_{A_i}(O_j) = \frac{1}{|A_i| - 1} \sum_{\substack{O_j, O_q \in A_i \\ O_j \neq O_q}} \left| \beta_{A_i}(O_j) - \Gamma(O_j, O_q) \right| \tag{3}$$

$$\eta_{A_k}(O_j) = \sum_{\substack{q=1 \\ i \neq q}}^{n} (1 - \Gamma(O_q^r, O_j)) \tag{4}$$

$\Gamma(O_j, O_q)$ is the similarity between objects O_j and O_q, O_q^r is the representative object in cluster q and n is the number of clusters. $\beta_{A_i}(O_j)$ is the average similarity of O_j with the other objects in the same cluster A_i. The $\alpha_{A_i}(O_j)$ function evaluates the variance between $\beta_{A_i}(O_j)$ and the similarity between O_j and the other objects in A_i and $\eta_{A_k}(O_j)$ is the average dissimilarity of O_j with the other representative objects.

The most representative object O_{Ri} in A_i must be the most similar in average with other objects in the cluster and the most dissimilar with respect to the other representative objects. These properties directly depend on $\beta_{A_i}(O_j)$ and $\eta_{A_k}(O_j)$ values respectively then O_{Ri} is that object that maximizes the expression $r_{A_i}(O_j)$.

The *MOSC* method (figure 3.1) starts creating n clusters. Once the clusters have been obtained, for each cluster A_j it is necessary to decide whether A_j is homogeneous or not.

If the cluster A_j is non homogeneous then A_j contains some objects located at critical regions, that is, border objects. In order to find the border objects, *MOSC* finds the majority class objects. Once these objects have been found, the border objects in the majority class are those objects that are the most similar to an object in A_j belonging to different class, and by analogy, the border objects of A_j in the other classes are those objects that are the most similar to each border object in the majority class.

If the cluster A_j is homogeneous then the objects in A_j are interior objects and they could be discarded from T without affecting the classification accuracy. Therefore, *MOSC* finds the representative object of the homogeneous cluster A_j and discards the remaining objects so that A_j is reduced to its representative object.

The objects selected by *MOSC* are the representative objects from each homogeneous cluster and the border objects from each non homogeneous cluster.

MOSC (Training set T, number of clusters n): object set S
$S = \varnothing$
Clust= *kMSF*(T, n) // create n clusters from T
For each cluster A_j in *Clust*
  If A_j is non homogeneous then
    Find the majority class C_M in cluster A_j
    For each class C_k in A_j ($C_k \neq C_M$)
      For each object O_j belonging to class C_k
        Find $O_c \in C_M$, the most similar object to O_j with class C_M
        $S = S \cup \{ O_c \}$
        Find O_M , the most similar object to O_c with class different to C_M
        $S = S \cup \{ O_M \}$
  Else // A_j is homogeneous
    O_i = representative object of the cluster A_j
    $S = S \cup \{ O_i \}$
Return S

Fig. 3.1. *MOSC* method for object selection

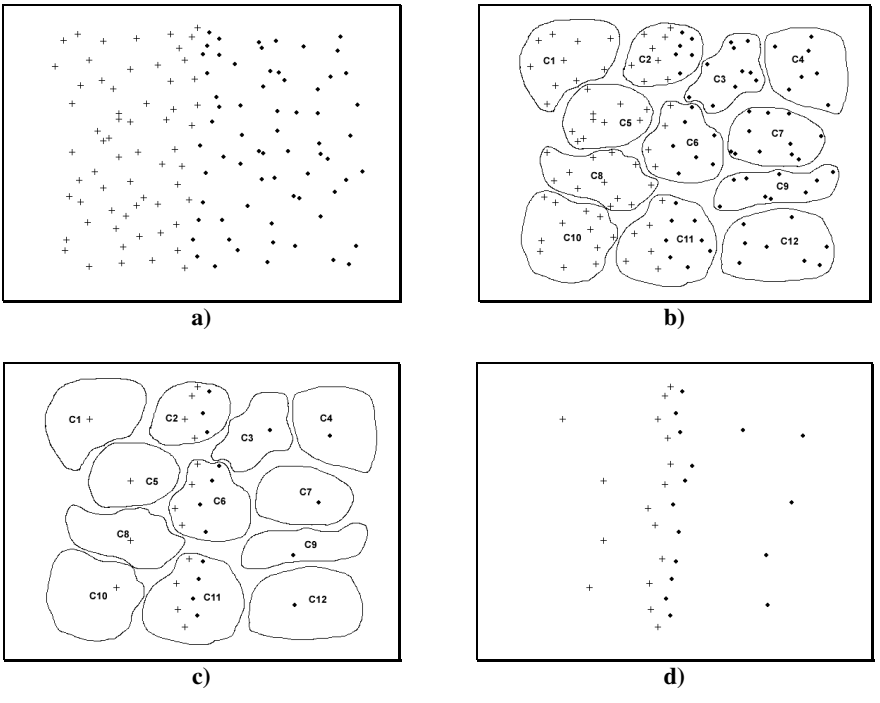

Fig. 3.2. a) Dataset with classes "+" and "•". **b)** Clusters created from the dataset. **c)** Objects selected in each cluster. **d)** Objects set obtained by *MOSC*.

To illustrate in a graphical way how *MOSC* finds border objects let us consider the dataset shown in figure 3.2a which is a bi-dimensional dataset created by hand with objects belonging to the classes "+" and "•". In figure 3.2b, the clusters (C1…C12) created from the dataset are depicted, the non homogeneous clusters are C2, C6 and C11 whereas the remaining clusters are homogeneous.

In the clusters C6 and C11 the minority class is "+", then the border objects in the most frequent class (•) are the most similar objects to each minority class object (+). On the other hand, the border object in class "+" are the most similar objects (belonging to class "+") to each border object in class "•".

The same process described before is applied to cluster C2 where the minority class is "•". The objects selected in each cluster are depicted in figure 3.2c and the objects set obtained by *MOSC* is depicted in figure 3.2d. We can observe that *MOSC* finds border objects and some interior objects (representative objects in the homogeneous clusters).

3 Experimental Results

In this section, we report the results obtained applying the *MOSC* method over ten datasets from the UCI dataset repository [10], four of them (*Glass, Iris, Liver, Wine*) are full numeric and the other six datasets are mixed. For all the experiments 10-fold cross validation is reported.

We show a performance comparison among *MOSC, CLU* and the *DROP* methods because according to the results reported in [4, 5], the *DROP* methods outperform to other relevant object selection methods such as *ENN, RENN* and *ICF*. We also compare against *CLU* because it is also an object selection method based on clustering.

For *MOSC* and *CLU* it is necessary to generate n clusters, $n \geq c$, where c is the total number of classes in the dataset. For these methods, we used the *k-means with similarity functions* algorithm for creating clusters.

In this work we used the next similarity function:

$$\Gamma(O_j, O_q) = 1 - \frac{HVDM(O_j, O_q)}{m} \tag{5}$$

Where *HVDM* (*Heterogeneous Value Difference Metric*) [4] is the function used and proposed by the *DROP*s authors and m is the number of features.

In order to choose the number of clusters n to be used in our experiments, we carried out an experiment over ten datasets using different values for n to choose the best ones where *MOSC* and *CLU* had the best performance in the average case. In table 4.1 we show the classification accuracy obtained by *MOSC* and *CLU* using the values $n=$ *2c, 4c, 6c, 8c* and *10c*. For testing the object sets selected by *MOSC* and *CLU*, the *k-Most Similar Neighbor* (*k-MSN*) classifier ($k=3$) was used, that is, *k-NN* but using a similarity function instead of a distance function for comparing objects. Also we show the classification obtained by the original training set (*Orig.*).

According to the results shown in table 4.1, the best value for n using *MOSC* was $n=8c$ and the best one for *CLU* was $n=10c$, therefore these values were used in all the experiments reported in the next tables.

Table 4.1. Classification accuracy obtained by *CLU* and *MOSC* using different number of clusters

Dataset		Number of clusters									
		n=2c		*n=4c*		*n=6c*		*n=8c*		*n=10c*	
	Orig.	**CLU**	**MOSC**	**CLU**	**MOSC**	**CLU**	**MOSC**	**CLU**	**MOSC**	**CLU**	**MOSC**
Bridges	66.09	46.09	45.45	51.63	51.63	53.54	56.54	58.36	59.45	61.27	61.09
Echocardiogram	95.71	89.82	94.46	85.90	86.42	90.71	86.42	94.10	91.42	90.71	85.53
Glass	71.42	42.85	54.71	50.45	62.27	55.64	64.04	61.29	64.48	62.00	63.52
Heart Cleveland	82.49	73.00	69.98	74.61	71.26	73.00	73.27	75.29	72.26	76.33	74.00
Heart Swiss	93.72	84.61	73.07	69.23	67.69	84.61	83.91	74.88	79.55	84.61	86.21
Hepatitis	79.29	79.25	77.25	77.50	73.12	75.00	75.37	75.87	79.29	73.66	75.54
Iris	94.66	64.64	90.66	89.33	92.66	88.66	91.33	91.33	94.66	90.00	90.00
Liver	65.22	55.03	57.98	53.94	59.68	48.19	59.40	46.40	59.40	51.89	59.15
Wine	94.44	73.33	86.66	88.88	88.88	92.22	94.44	90.00	91.11	94.44	94.44
Zoo	93.33	76.66	84.44	84.44	90.00	91.11	92.22	90.00	90.00	90.00	91.11
Average	**83.64**	68.53	73.47	72.59	74.36	75.27	77.69	75.75	78.16	77.49	78.06

In table 4.2 we report the results obtained by *DROP3, DROP5* (the best *DROP* methods reported in [4]), *CLU* and *MOSC* over the ten datasets. For each method we show the classification accuracy (*Acc.*) and the percentage of the original training set that was retained by each method (*Str.*), that is, 100*|*S*|/|*T*|. In addition, we show the classification obtained by the original training set (*Orig.*). The classifier used was *k-MSN* with *k=3* (the value of *k* reported in [4] for *DROP* methods, using *k-NN*). At the bottom of each table we show the average accuracy and storage obtained by each method.

Table 4.2. Classification (*Acc.*) and retention (*Str.*) results obtained using: the original training set (*Orig.*), *DROP3, DROP5, CLU* and *MOSC*

Dataset	Orig.		DROP3		DROP5		CLU		MOSC	
	Acc	*Str.*	*Acc*	*Str.*	*Acc*	*Str.*	*Acc*	*Str.*	*Acc*	*Str.*
Bridges	66.09	100	56.36	14.78	62.82	20.66	61.27	63.68	59.45	51.79
Echocardiogram	95.71	100	92.86	13.95	88.75	14.87	90.71	30.03	91.42	23.87
Glass	71.42	100	66.28	24.35	62.16	25.91	62.00	31.15	64.48	48.33
Heart Cleveland	82.49	100	78.89	11.44	79.87	14.59	76.33	18.33	72.26	26.21
Heart Swiss	93.72	100	93.72	1.81	93.72	1.81	84.61	18.06	79.55	15.89
Hepatitis	79.29	100	78.13	11.47	75.42	15.05	73.66	14.33	79.29	10.46
Iris	94.66	100	95.33	15.33	94.00	12.44	90.00	22.22	94.66	25.48
Liver	65.22	100	67.82	26.83	63.46	30.59	51.89	6.44	59.40	46.44
Wine	94.44	100	94.41	15.04	93.86	10.55	94.44	37.03	91.11	34.69
Zoo	93.33	100	90.00	20.37	95.56	18.77	90.00	76.41	90.00	50.24
Average	**83.64**	100	81.38	15.54	80.96	16.52	77.49	31.77	78.16	33.34

In figure 4.1, the classification (horizontal axis) versus retention (vertical axis) scatter graphic from results shown in table 4.2 is depicted. On this kind of graphic, the most located at right the best classification accuracy and the most located at bottom the best retention percentage.

Based on the results in table 4.2 and figure 4.1, we can observe that in the average case, the best object selection methods were *DROP3* and *DROP5*. The classification accuracy obtained by *MOSC* and *CLU* were smaller than those obtained by *DROPs* but *OSC* outperformed *CLU*.

The best methods in table 4.2 were the *DROPs* since the classifier was *k-MSN* and the *DROP*s are based on the Nearest Neighbor or Most Similar Neighbor rules, however it is important to test the object sets selected (obtained in the previous experiment by *DROPs, CLU* and *MOSC*) as training sets for other classifiers, in particular we are interested in testing with classifiers that allow handling mixed data.

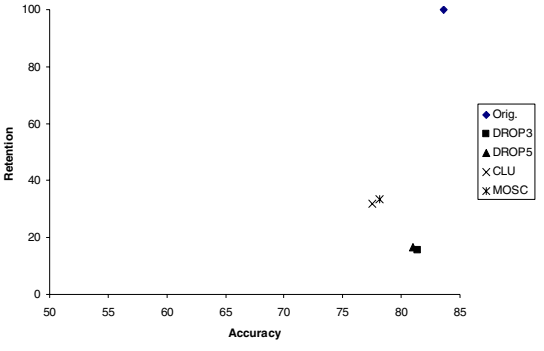

Fig. 4.1. Scatter graphic from results shown in table 4.2

Therefore, another experiment was done using the object set obtained by *DROPs, CLU* and *MOSC* as training sets for the *C4.5* [11] and *ALVOT* [12] classifiers, which allow handling mixed data. For *ALVOT,* we used as support sets system all the features subsets with cardinality 3. The row evaluation function for a fixed support set Ω was:

$$\Gamma_{\Omega}(O_p, O) = \beta_{\Omega}(O_p, O) \tag{6}$$

Where $\Omega \in \Omega_A$, Ω_A is the support set system, and $\beta(O_p, O)$ is the similarity function shown in (5) but comparing only the features of Ω.

The class evaluation function for a fixed support set Ω was:

$$\Gamma_{\Omega}^{j}(O) = \frac{1}{m_j} \sum_{t=1}^{m_j} \Gamma_{\Omega}(O, O_t) \tag{7}$$

Where m_j is the number of objects in the *j*-th class.

The evaluation by class for the whole support set system Ω_A was obtained using the next expression:

$$\Gamma_j(O) = \frac{1}{|\Omega_A|} \sum_{\Omega \in \Omega_A} \Gamma_{\Omega}^{j}(O) \tag{8}$$

Finally, a new object is assigned to the class where it obtains the higher evaluation.

The *C4.5* and *ALVOT* results are reported in tables 4.4-4.5 and figures 4.2-4.3 respectively.

Table 4.4. Classification results obtained using the original training set (*Orig.*) and the object sets obtained by *DROPs, CLU* and *MOSC* as training for the *C4.5* classifier

Dataset	Orig.	DROP3	DROP5	CLU	MOSC
Bridges	65.81	47.90	39.54	55.45	59.17
Echocardiogram	95.71	84.10	92.85	93.21	95.89
Glass	67.29	60.19	53.76	58.35	62.22
Heart Cleveland	71.96	68.59	72.16	76.57	73.59
Heart Swiss	93.71	93.71	93.71	84.61	82.81
Hepatitis	76.70	63.33	63.41	71.58	65.68
Iris	93.99	92.66	90.66	92.66	93.99
Liver	63.67	59.48	63.67	57.96	61.11
Wine	94.44	84.43	78.88	86.65	86.65
Zoo	93.33	81.10	88.88	92.21	93.33
Average	81.66	73.55	73.75	76.93	77.44

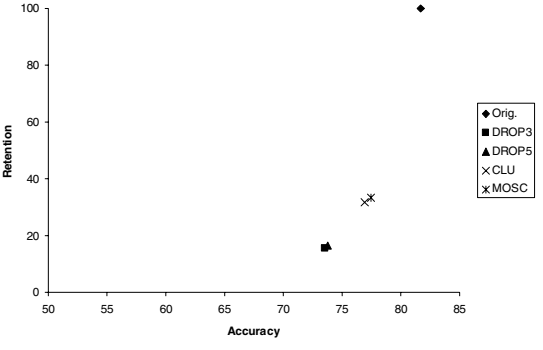

Fig. 4.2. Scatter graphic from results shown in table 4.4

Table 4.5. Classification results obtained using the original training set (*Orig.*) and the object sets obtained by *DROPs, CLU* and *MOSC* as training for the *ALVOT* classifier

Dataset	Orig.	DROP3	DROP5	CLU	MOSC
Bridges	22.81	23.09	27.27	22.81	20.00
Echocardiogram	93.21	93.21	90.35	93.21	87.50
Glass	40.56	29.95	28.09	40.90	40.56
Heart Cleveland	72.59	73.26	73.89	73.26	72.30
Heart Swiss	66.53	76.23	76.23	76.21	76.00
Hepatitis	81.12	35.12	41.87	24.08	45.58
Iris	86.66	88.66	88.66	87.33	88.66
Liver	48.44	48.57	54.77	48.07	48.44
Wine	90.00	83.69	89.86	92.22	92.22
Zoo	96.66	90.00	84.44	96.66	96.66
Average	69.86	64.18	65.54	65.48	66.79

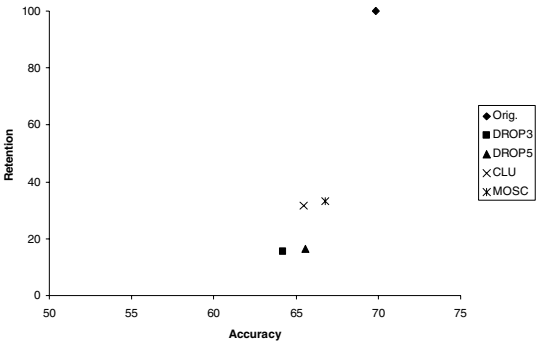

Fig. 4.3. Scatter graphic from results shown in table 4.5

Based on the results shown in tables 4.4 and figure 4.2, in the average case, for *C4.5*, the best object selection method was *MOSC* followed by *CLU*, that is, in this experiment, the subsets obtained by the *DROP*s were not as good as the obtained by *MOSC*.

According to results shown in table 4.5 and figure 4.3, again (as in table 4.4), in the average case for *ALVOT*, the best method was *MOSC* followed by *DROP5* and *CLU* respectively. Notice that in table 4.5, there are some low accuracy results; this is due to the *ALVOT* sensitivity to imbalanced classes.

4 Conclusions

The object selection is an important task for instance-base classifiers since through this selection the times in training and classification could be reduced. In this paper we proposed and compared the *MOSC* object selection method based on clustering. This method finds some interior and border objects since through these last it is possible to preserve discrimination capability between classes in a training sample. In addition, *MOSC* allows handling not only numeric but also nominal data which is useful since in practice it is very common to face with mixed data problems.

The experimental results showed that *MOSC* is a good method for solving the object selection problem when a classifier different from *k-MSN* is used. Since most of the object selection methods follow the nearest or most similar neighbor rule, the object sets obtained by these methods have not a good performance when they are used as training for other classifiers which are not based on the nearest or most similar neighbor rules, as it can be seen in our experimental results. These results showed that the objects sets selected by *MOSC* had a better average performance when they are used as training for the *C4.5* and *ALVOT* classifiers.

As future work, we will do experiments using other mixed data clustering methods and we will propose another way for selecting border objects in non homogeneous clusters.

References

1. Hart, P.E.: The Condensed Nearest Neighbor Rule. IEEE Transactions on Information Theory 14(3), 515–516 (1968)
2. Wilson, D.L.: Asymptotic Properties of Nearest Neighbor Rules Using Edited Data. IEEE Transactions on Systems, Man, and Cybernetics 2(3), 408–421 (1972)
3. Tomek, I.: An Experiment with the Edited Nearest-Neighbor Rule. IEEE Transactions on Systems, Man, and Cybernetics 6(6), 448–452 (1976)
4. Wilson, D.R., Martínez, T.R.: Reduction Techniques for Instance-Based Learning Algorithms. Machine Learning 38, 257–286 (2000)
5. Brighton, H., Mellish, C.: Advances in Instance Selection for Instance-Based Learning Algorithms. Data Mining and Knowledge Discovery 6, 153–172 (2002)
6. Leung, Y., Zhang, J.S., Xu, Z.B.: Clustering by scale-space filtering. IEEE Trans. on Pattern Analysis and Machine Intelligence 22, 1396–1410 (2000)
7. Spillmann, B., Neuhaus, M., Bunke, H., Pękalska, E., Duin, R.P.W.: Transforming Strings to Vector Spaces Using Prototype Selection. In: Yeung, D.-Y., Kwok, J.T., Fred, A., Roli, F., de Ridder, D. (eds.) Structural, Syntactic, and Statistical Pattern Recognition. LNCS, vol. 4109, pp. 287–296. Springer, Heidelberg (2006)
8. Lumini, A., Nanni, L.: A clustering method for automatic biometric template selection. Pattern Recognition 39, 495–497 (2006)
9. García-Serrano, J.R., Martínez-Trinidad, J.F.: Extension to C-means algorithm for the use of similarity functions. In: Żytkow, J.M., Rauch, J. (eds.) Principles of Data Mining and Knowledge Discovery. LNCS (LNAI), vol. 1704, pp. 354–359. Springer, Heidelberg (1999)
10. Blake, C., Keogh, E., Merz, C.J.: UCI repository of machine learning databases. In: Department of Information and Computer Science, University of California, Irvine, CA (1998), http://www.ics.uci.edu/ mlearn/MLRepository.html
11. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco (1993)
12. Ruiz-Shulcloper, J., Abidi, M.A.: Logical Combinatorial Pattern Recognition: A Review. In: Pandali, S.G. (ed.) Recent Research Developments in Pattern Recognition. Transword Research Networks, USA, pp. 133–176 (2002)