CENATAV

Centro de Aplicaciones de Tecnologías de Avanzada

MINISTERIO DE LA INDUSTRIA BÁSICA

RNPS No. 2142
ISSN 2072-6287
Versión Digital

REPORTE TÉCNICO
Reconocimiento de Patrones

SERIE AZUL

Prototype Selection Methods for Dissimilarity Space Classification

Lic. Yenisel Plasencia Calaña,
Dr. C. Edel García Reyes, Dr. ir. Robert P. W. Duin, Dr. Ing. Mauricio Orozco-Alzate

RT_027 **marzo 2010**

7ma. No. 21812 e/218 y 222,
Rpto. Siboney, Playa;
Ciudad de La Habana.
Cuba. C.P. 12200
www.cenatav.co.cu

CENATAV
Centro de Aplicaciones de Tecnologías de Avanzada
MINISTERIO DE LA INDUSTRIA BÁSICA

RNPS No. 2142
ISSN 2072-6287
Versión Digital

SERIE AZUL

REPORTE TÉCNICO
Reconocimiento de Patrones

Prototype Selection Methods for Dissimilarity Space Classification

Lic. Yenisel Plasencia Calaña,
Dr. C. Edel García Reyes, Dr. ir. Robert P. W. Duin, Dr. Ing. Mauricio Orozco-Alzate

RT_027 **marzo 2010**

7ma. No. 21812 e/218 y 222,
Rpto. Siboney, Playa;
Ciudad de La Habana.
Cuba. C.P. 12200
www.cenatav.co.cu

Index

Prototype Selection Methods for Dissimilarity Space Classification

Lic. Yenisel Plasencia[12], Dr. C. Edel García-Reyes[1], Dr. ir. Robert P. W. Duin[2], Dr. Ing. Mauricio Orozco-Alzate[3]

[1] Centro de Aplicaciones de Tecnología de Avanzada, 7a #21812 e/ 218 y 222, Siboney, Playa, Ciudad de La Habana, Cuba
{yplasencia,egarcia}@cenatav.co.cu
[2] Faculty of Electrical Engineering, Mathematics and Computer Sciences, Delft University of Technology, The Netherlands
r.duin@ieee.org
[3] Departamento de Informática y Computación, Universidad Nacional de Colombia Sede Manizales Campus La Nubia, km 7 vía al Magdalena. Manizales, Colombia
morozcoa@unal.edu.co

RT_ 027 CENATAV
Fecha del camera ready: 30 de octubre de 2009

Abstract. A common way to represent patterns for recognition systems is by vectors lying in a vector space. As this representation is based only on the object features, there is no need to have more objects to construct it. In contrast, a dissimilarity representation of patterns takes into account the relations between them by some measure of resemblance (e.g. dissimilarity). The nearest neighbour (1-NN) is a dissimilarity-based classifier that has shown to be very competitive for several Pattern Recognition problems. Classification results on dissimilarity spaces spanned by dissimilarities to prototypes can reach or improve the 1-NN results in terms of accuracy and computational efficiency. This is possible if a small set of prototypes is selected with similar or better discriminative power than the complete set of initial prototypes. How to obtain this set has been studied by researchers in the area of Dissimilarity Representations and Graph Representations by means of Prototype Selection Methods. This report presents a description and analysis of different approaches published on this interesting topic.

Keywords: Prototype Selection, Dissimilarity Space

Resumen. Una manera usual de representar patrones para sistemas de reconocimiento es mediante vectores que yacen en un espacio vectorial. Como esta representacion se basa solo en las características del objeto, no es necesario tener más objetos para construirla. En contraste, una representación por disimilitud de los patrones toma en cuenta las relaciones entre ellos mediante alguna medida de semejanza (ejemplo disimilitud). El vecino más cercano (1-NN, por sus siglas en ingles) es un clasificador basad en disimilitud que ha mostrado ser muy competitivo para muchos problemas de reconocimiento de patrones. Los resultados de clasificación en espacios de disimilitud generados por disimilitudes a prototipos pueden alcanzar o mejorar los resultados del 1-NN en términos de precisión y de eficiencia computacional. Esto es posible si un conjunto pequeño de prototipos es seleccionado con poder discriminativo similar o mejor que el del conjunto completo de prototipos iniciales. Cómo obtener este conjunto ha sido estudiado por investigadores en el área de representación por disimilitudes y representación por grafos mediante métodos de selección de prototipos. Este reporte presenta una descripción y análisis de diferentes enfoques publicados en este interesante tema.

Palabras clave: selección de prototipos, espacio de disimilitud

1. Introduction

Pattern Recognition is a wide research area closely related to many other fields such as statistics, algebra, computer science, machine learning, image processing, etc. Most of the studies in this area are concentrated in the stages of data preparation or preprocessing, data representation and classification methods. Data preprocessing includes methods for standardizing the data, that is dealing with invariants for the problem at hand. For example in a face recognition system this can include face geometric normalization, registration with respect to a template, face cropping and background removal, etc. Depending of each specific problem and methods, the preprocessing can be skipped and invariants can be incorporated in the object representation directly. If this representation does not contain the object features that are good for discriminating between classes in some manner; it will be difficult to design optimal classifiers for the recognition problem. Also, focusing on improving data representation can lead to an improvement on classification performances.

Three major groups of methods can be distinguished for representation in the Pattern Recognition literature. The first group represents objects as vectors in Euclidean vector spaces. The second represents the object structure by graphs, grammars etc. The third group represents the objects by their nearness relation with respect to other objects.

The vectorial representation encodes the objects as vectors of numbers of fixed dimension, and assumes that this vectors lie in a Euclidean or metric vector space. In practice this does not hold for all problems, but once it is assumed, a plethora of statistical tools can be used. However, if it is wanted to incorporate expert knowledge and invariance to the representation, the Euclidean geometry is a very strong assumption that may not be fulfilled and totally new representation approaches should be developed.

The structural approach [1] focuses in modeling the parts of the object and the relations between them in a more intuitive way, and seems to be more robust than the vectorial one, but the tools for handling this structural representation are not as developed as the tools that can be used for data lying in Euclidean vector spaces.

Pekalska et al [2] proposed the dissimilarity representation approach. Intuitively, the nearness information is more important for discriminating between the classes than the composition and features of each object independently. Also, this approach has the potential of unifying the statistical and the structural approach; because for example dissimilarities can be computed from a structural representation. In [2] we can find theory, methods, experimental results and open questions on the dissimilarity representation [1].

There are three different approaches for classification using a dissimilarity representation:

1. K nearest neighbour (k-nn) rule applied to the dissimilarity matrix.
2. Classifiers constructed in dissimilarity spaces.
3. Classifiers constructed in embedding spaces.

In this study we will focus on the second approach. Its angular stone is the postulation of the dissimilarity space that is briefly presented in the next section.

2. Dissimilarity Space

The dissimilarity space was proposed by Pekalska et al.[2]. It was postulated as a Euclidean vector space. For its construction a representation set $R = \{r_1, r_2, ..., r_n\}$ is needed, where the objects belonging to this set (also called prototypes) are chosen adequately based on some criterion that can be dependant on the problem at hand. Let X be the training set, R and X can have the following relationships: $R \cap X = \varnothing$, or $R \subseteq X$. Once we have R, the dissimilarities of the objects in X to the objects in R are computed. When a new object r comes, it is also represented by a vector of dissimilarities d_r to the objects in R (1).

$$d_r = [d(r, r_1)d(r, r_2)...d(r, r_n)]. \tag{1}$$

The dissimilarity space is defined by the set R so each coordinate of a point in that space corresponds to a dissimilarity to some prototype and the dimension of this space is determined by the amount of prototypes selected. This allows us to control the computational cost and to guarantee the trade off between classification accuracy and computational efficiency. Sometimes, a smaller representation set can lead to better classification results than a larger one, e.g. when we discard noisy objects for prototypes. Different Prototype Selection methods have been developed with the aim to find a small representation set that is still capable of generating a dissimilarity space where classifiers can discriminate between the classes as well as or better than with all the initial objects.

3. Prototype Selection Methods

For dissimilarity representations the adaptation of prototype selection techniques available for the vector space representation or feature-based approach has been investigated showing good results [3]. Also new techniques have been investigated [4].

In general in the k-NN literature two basic types of algorithms can be identified: prototype generation and prototype selection [4,2]. The first group focuses on merging the initial prototypes in a way that optimizes the performance of k-NN. Examples of these algorithms are Kmeans [5] and LVQ [6]. The second group focuses on reducing the original set. Condensing methods identify a small set so that the overall performance in this set is similar to the performance in the original set. Editing methods remove noisy samples leaving smooth decision boundaries [7,8]. Generally, condensing methods are applied after the editing methods. The editing and condensing methods have the disadvantage of usually working in Euclidean spaces.

The main difference of the application of prototype selection methods for k-NN and for dissimilarity space classifiers is that in the first case, the techniques are applied for choosing the final training objects; and in the second case for determining the prototypes to construct the dissimilarity space, since still all the initial objects will be used for training. In [3] the authors compared prototype selection methods for classification in dissimilarity spaces showing good results when used with linear and quadratic classifiers. In [4] various techniques were compared such as Kcentres, Modeseek, feature selection, linear programming, editing-condensing methods, and a mixture of Kcentres with linear programming.

These techniques showed good performance especially for small sets of prototypes, where random selection performed worse. Other prototype selection methods have been proposed in the graph and string domain [9,10]. The methods tackle the question of how to select a small representation set for constructing the dissimilarity space.

It is of our interest to detect the methods that can be applicable also to generalized dissimilarity representations, so instead of selecting objects for prototypes the methods will select models of the objects as prototypes. We can find studies on generalized dissimilarity representations using feature lines and feature planes [11].

This report briefly summarizes some of the studies presented in the literature on prototype selection methods applicable to dissimilarity data computed from the initial objects directly, from vectorial representations and from graphs or strings, where prototypes can be objects or models. Also, methods that create or generate new prototypes instead of selecting them from a set of objects or models that already exist have been investigated, but this methods are out of the scope of this report. Examples of those methods are: means of clusters [12], LVQ and mixture of gaussians [3]. For clarification and more details on the methods, the reader can refer to the original papers. Different aspects will be compared such as:

1. The applicability of the method to generalized dissimilarity representations.
2. If the method finds a fixed number k of prototypes (controllable by the user) or just returns an arbitrary number.
3. If the selection is per class (class-wise) or just in a bunch of data that can be the training set, a subset of it or with no overlap at all.
4. Dependance of the selected representation set on the initialization of the method.
5. If the method interprets the dissimilarities as such and not as arbitrary numbers, that is the fact that small numbers indicate high resemblance and large numbers indicate small resemblance is taken into account.
6. If the class separability is optimized.
7. Ability for dealing with non-metric data, that is if the dissimilarity measure is not restricted to be metric.

3.1. Random

This method selects k prototypes randomly from the training set [4]. It can work well for large prototype sets but if a small prototype set is needed, is better to go for a systematic approach. With this method it is possible to find redundant prototypes. Table 1 shows the features of the method for the aspects to compare.

Table 1. Random Selection

Aspect to compare	
Applicable to generalized Diss Rep	yes
Controllable or arbitrary number of prototypes	controllable
Per class	no
Dependent on an initialization	yes
Interprets dissimilarities	no
Class separability taken into account	no
Allows non-metric dissimilarity	yes

3.2. RandomC

The method [4] selects k prototypes from each class, in contrast to random selection of k objects without taking into account the class labels (table 2). This allows the distribution of prototypes to be more uniform with respect to the class distribution. If the classes are spread, it will work similar to the random selection.

Table 2. Random Selection per Class

Aspect to compare	
Applicable to generalized Diss Rep	yes
Controllable or arbitrary number of prototypes	controllable
Per class	yes
Dependent on an initialization	yes
Interprets dissimilarities	no
Class separability taken into account	no
Allows non-metric dissimilarity	yes

3.3. KCentres

The method is based on a clustering procedure for dissimilarity data [4]. It selects k prototypes from each class in a way that they are evenly distributed with respect to the dissimilarity information. Initially, a candidate set composed by k objects is chosen from each class randomly or in a more refined way. Then, for each object, its nearest neighbour from its class candidate set is computed. All the objects of the class that have the same nearest neighbour are grouped in one cluster. Then each cluster center object is found, that is the one who has the minimum distance with respect to its cluster objects. If one of the objects that initially belonged to the candidate set of the class is replaced by one new object as center of its cluster, the procedure is repeated until no replacement takes place. The final prototype set is obtained by joining the prototype sets computed from each class. This procedure has the disadvantage of being dependant of the initial candidate sets (table 3).

Table 3. KCentres

Aspect to compare	
Applicable to generalized Diss Rep	no
Controllable or arbitrary number of prototypes	controllable
Per class	yes
Dependent on an initialization	yes
Interprets dissimilarities	yes
Class separability taken into account	no
Allows non-metric dissimilarity	yes

3.4. Modeseek

The method is based also on a clustering procedure for dissimilarity data [4]. As its name suggests, the method finds the modes of the data. It selects a number of prototypes per class, but this number cannot be set by the user as in the case of the previous methods. Instead, a neigbourhood parameter which helps to control the number of prototypes should be provided. Let us suppose that this parameter is s. For each class object x_j, the method finds the dissimilarity to its sth neighbour. Then, for the s neighbours of x_j, the dissimilarities to their sth neighbours are also computed. If the dissimilarity of x_j to its sth neighbour is minimum compared to those of its s neighbours, it is selected as prototype. As this process is executed per class, it is needed to join the resulting prototypes for each class to conform the final prototype set (table 4).

Table 4. ModeSeek

Aspect to compare	
Applicable to generalized Diss Rep	no
Controllable or arbitrary number of prototypes	controllable
Per class	yes
Dependent on an initialization	no
Interprets dissimilarities	yes
Class separability taken into account	no
Allows non-metric dissimilarity	yes

3.5. FeatSel

This procedure has one big difference with respect to the above mentioned ones. The difference is that this method optimizes the separability of the classes. Here, the prototypes are treated as features, and the prototype selection problem can be tackled as a feature selection problem with the dissimilarities to the prototypes as feature values. In the method proposed in [4], some modifications are made to the original forward feature selection. The features are considered in a dissimilarity space, but the classification error for each feature set, is calculated based on the leave-one-out 1-NN error on the original dissimilarity matrix and not in the dissimilarity space. In this case the nearest neighbour of one object will be the prototype who has the smallest dissimilarity to the object. The authors solve ties for different representation sets by selecting the one for which the sum of dissimilarities is minimum (table 5).

Table 5. FeatSel

Aspect to compare	
Applicable to generalized Diss Rep	yes
Controllable or arbitrary number of prototypes	controllable
Per class	no
Dependent on an initialization	no
Interprets dissimilarities	yes
Class separability taken into account	yes
Allows non-metric dissimilarity	yes

3.6. LinProg

This method [4] solves an optimization problem in order to train a separating hyperplane $w^T D(x, R) + w_0$ in a dissimilarity space $D(T, R)$. The w_j are expressed by non negative variables α_j and β_j as $w_j = \alpha_j - \beta_j$. In the optimization problem it is also introduced a nonnegative slack variable ξ_i that accounts for classification errors as well as a regularization parameter C. Let $x_i \in T$ be training objects with class labels $y_i \in 1, -1$, the minimization problem is formulated as follows.

$$\text{mín} \quad \sum_{i=1}^{n}(\alpha_i + \beta_i) + \gamma \sum_{i=1}^{n} \xi_i \tag{2}$$

$$\text{subject to} \quad y_i f(D(x_i, R)) \geq 1 - \xi_i, i = 1, ..., n \tag{3}$$

$$\alpha_i, \beta_i, \xi_i \geq 0. \tag{4}$$

The final prototypes are the objects that have w_j weights different from 0. The characteristics of the method are shown in table 6.

Table 6. LinProg

Aspect to compare	
Applicable to generalized Diss Rep	no
Controllable or arbitrary number of prototypes	arbitrary
Per class	no
Dependent on an initialization	no
Interprets dissimilarities	no
Class separability taken into account	yes
Allows non-metric dissimilarity	yes

3.7. EdiCon

This is the classical Editing Condensing algorithm applied to the original dissimilarity matrix [4] and not to a dissimilarity space. The Editing method removes those objects that are erroneously classified by the 1-NN, so the overlapping of classes is decreased. Then, Condensing is applied. Condensing removes objects taking care that the performance of the 1-NN classifier on the new set is similar to the performance using all the training objects. As it can be seen, the algorithm returns at least one prototype per class, since the 1-NN needs to have some pattern of each class to measure the resemblance of a new incoming test object (table 7).

Table 7. EdiCon

Aspect to compare	
Applicable to generalized Diss Rep	no
Controllable or arbitrary number of prototypes	arbitrary
Per class	no
Dependent on an initialization	yes
Interprets dissimilarities	yes
Class separability taken into account	yes
Allows non-metric dissimilarity	yes

3.8. Center Prototype Selector

This procedure was proposed in [10] for representing strings by edit distances and in [9] for representing graphs as it is also the case of the Border, Spanning, and K-Medians Prototype selectors that will be explained in the next subsections. The center prototype selection method starts from the set median string, and gradually adds the remaining median objects. Since they are in the center of the data, the selected prototypes can be redundant on their contribution for representing the rest of the objects, but on the other hand outliers are avoided, which is a desirable property in order to discard noise (table 8). The reader can get confused with the Kcentres prototype selection. There is a difference between the two of them. The Kcentres starts from a set of k objects determining k clusters, and every training object is assigned to the nearest cluster. So, every object is a potential candidate for prototype. In the Center Prototype selection only the most central objects from the training objects or models can be selected. In the Kcentres, objects in the border of the data distribution can be selected as prototypes, they only need to fulfill the property of being center of its cluster, and not of the training objects.

Table 8. Center Prototype Selector

Aspect to compare	
Applicable to generalized Diss Rep	yes
Controllable or arbitrary number of prototypes	controllable
Per class	no
Dependent on an initialization	no
Interprets dissimilarities	yes
Class separability taken into account	no
Allows non-metric dissimilarity	yes

3.9. Border Prototype Selector

As its name suggests, this method is based on selecting those objects belonging to the border of the data distribution [10]. It is important to emphasize that whether one object belongs to the center or border of the dissimilarity data is determined by the dissimilarity measure used and the problem at hand. It also depends on how the terms Çenter.ªnd "Border.ªre defined. In this method, the problem of the similar contribution of the center prototypes selected by the previous method is avoided. On the other hand, outliers are likely to be selected, since they usually are in the borders of the data distribution (table 9).

Table 9. Border Prototype Selector

Aspect to compare	
Applicable to generalized Diss Rep	yes
Controllable or arbitrary number of prototypes	controllable
Per class	no
Dependent on an initialization	no
Interprets dissimilarities	yes
Class separability taken into account	no
Allows non-metric dissimilarity	yes

3.10. Spanning Prototype Selector

Starting from the set median string, the method adds as prototype the object with largest distance to the actual representation set [10]. In this way, objects that yield similar contribution to the representation set are not likely to be selected, since the next prototype is always the farthest one. Also, the prototypes will tend to be uniformly distributed. The authors also point that outliers are likely to be selected since they have a large distance to other objects (table 10).

Table 10. Spanning Prototype Selector

Aspect to compare	
Applicable to generalized Diss Rep	no
Controllable or arbitrary number of prototypes	controllable
Per class	no
Dependent on an initialization	no
Interprets dissimilarities	yes
Class separability taken into account	no
Allows non-metric dissimilarity	yes

3.11. Length-based selection of feature lines

Feature lines are linear models of the objects constructed in a class wise manner [13,11]. Nevertheless, it is also possible to construct them without taking into account the class labels. The method is based on sorting all the lines by length. Then it is possible to start the selection by the one that has the smallest length, and gradually add the next always by smallest size. Another possibility is to select the set of lines that has the largest length. Selecting the middle length feature lines was also considered in order to describe slightly curved manifolds (table 11).

Table 11. Length-based selection of feature lines

Aspect to compare	
Applicable to generalized Diss Rep	yes (only for feature lines)
Controllable or arbitrary number of prototypes	controllable
Per class	yes
Dependent on an initialization	no
Interprets dissimilarities	yes
Class separability taken into account	no
Allows non-metric dissimilarity	no

3.12. MaxNCN

This is a condensing method based on the concept of Nearest Centroid Neighbours (NCN) [3]. For an object p, the nearest objects are the ones that are closer to the object. In addition, for the NCN, it is also considered the symmetrical distribution of the nearest objects around p. The idea is based on the assumption that the geometry of the distribution of objects may be more informative than the distances between them. The method proceeds as follows. The first NCN is the nearest neighbour of p. The next NCN to be added is the one that with the previously added NCN, determines the closest centroid to p. Objects found like this, tend to surround p. The process continues while the class of the next maxNCN is the same as the class of p. As prototypes of one class should be located in a neighbouring area, they can be replaced by a single prototype without a major loss in their representation potential. The first prototype for a class is the object with a larger number of NCN. The algorithm removes the NCNs of this prototype and the prototype itself, and updates the number of neighbours of the remaining class objects as being part of an already processed group or neighbourhood. Then, again the object with larger number of NCN is selected, until it is not possible anymore to select a new prototype (table 12).

Table 12. MaxNCN

Aspect to compare	
Applicable to generalized Diss Rep	yes
Controllable or arbitrary number of prototypes	controllable
Per class	yes
Dependent on an initialization	no
Interprets dissimilarities	yes
Class separability taken into account	no
Allows non-metric dissimilarity	no

3.13. Reconsistent

The MaxNCN procedure has the disadvantage of discarding objects that are close to the decision boundary. The Reconsistent method tries to tackle this problem. After the MaxNCN procedure is applied, all the objects from the training set are classified by the 1-NN with respect to the prototype set obtained by the MaxNCN. The set of misclassified objects is condensed taking as reference the prototype set. Then, the condensed set is added to the prototype set returned by MaxNCN in order to conform the final representation set. In spite of the fact that the method does not take class separability into account by using the minimization of a classification error, somewhat when it stops once it is reached one object of a different class, this can be seen a a way of considering the class separability (table 13).

Table 13. Reconsistent

Aspect to compare	
Applicable to generalized Diss Rep	yes
Controllable or arbitrary number of prototypes	controllable
Per class	yes
Dependent on an initialization	no
Interprets dissimilarities	yes
Class separability taken into account	no
Allows non-metric dissimilarity	no

4. Concluding remarks

It is of the author's interest to expose to the reader the characteristics of current state of the art Prototype Selection Methods for Dissimilarity Space Classification in order to help to decide the ones that can be used for a particular problem and/or setup. For example, if the expert comes with a classification problem in terms of dissimilarity data, and he knows in advance that the dissimilarity matrix do not fulfill the metric requirements, then the MaxNCN cannot be a choice. This method and the Reconsistent were created for data lying in a metric vector space, so they are not flexible enough to allow non-metric data. The fact that most of the methods can be applicable to non-metric data, allows the use of a dissimilarity function or data created by the experts exploiting knowledge on their specific domains, that should be more robust and informative than one general metric such as the Euclidean.

Prototype selection methods that allow the user to control the number of prototypes to obtain are more adequate for developing systems with execution time constrains, but if it is wanted to obtained a small representation set and as informative as when using all the initial set, the methods that automatic select the optimal number of prototypes may be preferred.

For the case when the methods can be executed in a class-wise way, one advantage is that uniformity of the prototypes distribution can be gained when the classes are evenly spread and balanced. If classes are unbalanced, finding a fixed number of prototypes per class will not represent the true distribution, instead the number of prototypes should be proportional to the number of elements of each class. Also, exploiting class labels can be beneficial since more information of the problem is being used. One interesting question is whether it is really necessary to have prototypes from all the classes. Maybe with prototypes of a few classes it is possible to describe our data without loosing classification accuracy. When class labels are not provided, class-wise prototype selection methods are no longer adequate.

The dependency on initialization is a problem that researchers have been trying to solve by putting effort in finding a good initial set. The solutions of prototype selection methods that depend on initialization are not stable for different runs of the algorithm, as in the case of other type of procedures that suffer from this issue. For the Kcentres procedure [4] the authors took care of finding carefully the initial set, and not just randomly.

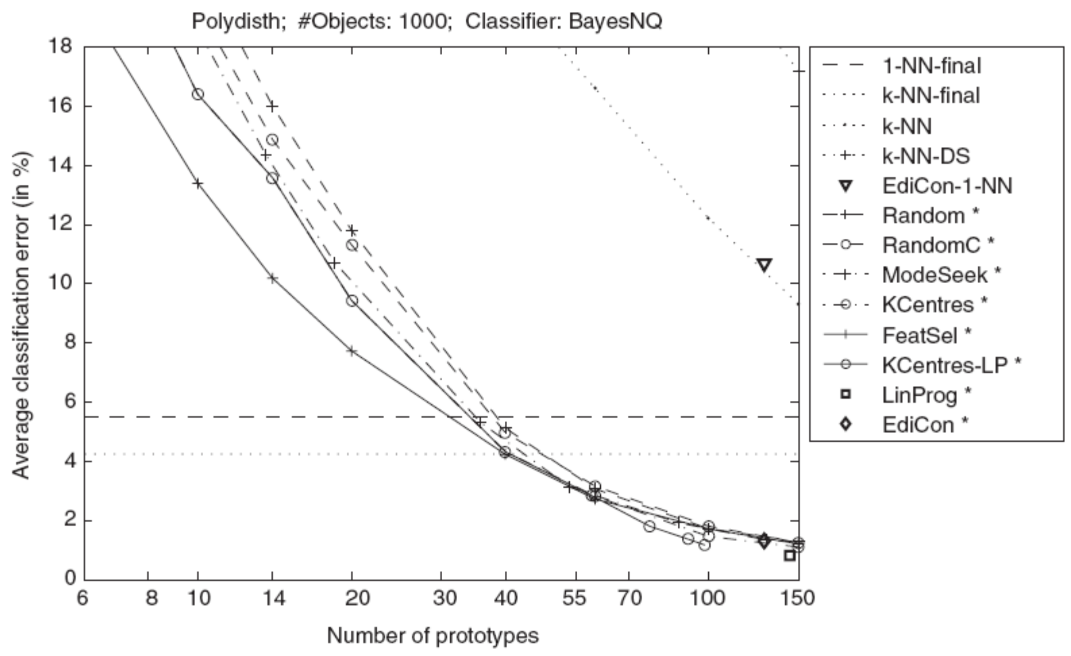
Generalized dissimilarity representations allow more flexibility than dissimilarity representations based on just object distances. The possible applicability of the already published methods for dissimilarity representations to generalized ones is information that researchers in the area can take into account as a baseline for their studies. Also, the homogeneity achieved when having the possibility to test the same method in both representations, can help to understand better under which circumstances one representation should be preferred.

When class separability can be taken into account in the selection criteria as in the FeatSel case, we should be able to obtain good classification performances with a small representation set. This was showed in [4], where the FeatSel method outperformed the others in various data sets. This can be seen in Fig. 1, where the original results are borrowed from the authors for illustration of this aspect to the reader.

The methods are also compared in terms of interpretation of the dissimilarities or lack of it. This aspect tell us if dissimilarities are taken just as arbitrary numbers by the algorithms or if they take into account that in this numbers the resemblance and proximities between the objects is measured. The fact of interpreting this values can make the methods more robust since it is extra knowledge to be exploited and prototype selection methods can benefit from it.

In conclusion, Random, RandomC and Kcentres have the disadvantage of being dependent on the initialization. Kcentres and ModeSeek interprets dissimilarities but cannot be applicable to generalized dissimilarity representations. FeatSel and EdiCon are the only methods that optimize class separability on the training set. The Center Prototype Selector returns prototypes with redundant contribution and the Border and Spanning Prototype Selector have the disadvantage of returning outliers as prototypes. MaxNCN and Reconsistent needs to have a feature representation available in order to compute the centroids, also they do not allow the use of non-metric dissimilarities.

Results in [4] demonstrate that, generally, the 1-NN classification performances can be reached or outperformed by using prototype selection methods with a small representation set, implying a lower computational cost. There was no "superior"prototype selection method, it seems that this depends on the problem at hand, dissimilarity measure used, and data distribution. Also, systematic procedures outperformed random ones.
For visualization purposes, plots of the prototype selection results for some of the algorithms will be shown. The figures show 20 prototypes out of 400 objects for the methods



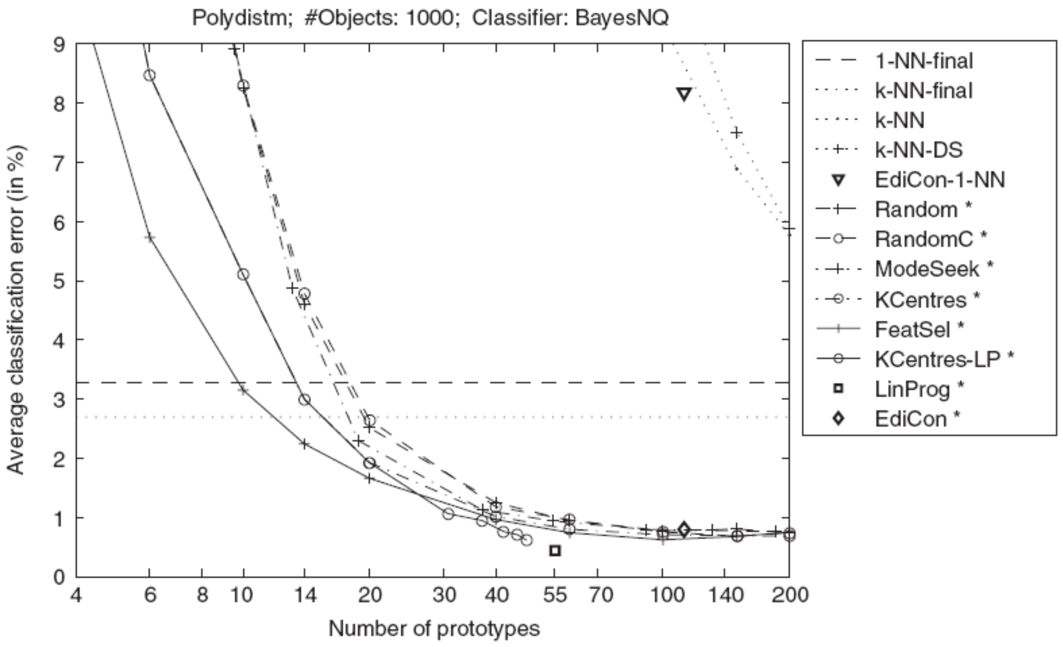


Figure 1. FeatSel compared to other prototype selection methods on the Polygon data [4]

that allow to set the number of prototypes by the user. It was plotted the 2D Principal Component projection for a 9 class classification problem. Class labels were omitted in order to emphasize the positions of prototypes that are plotted using a red color. Fig. 2 shows the results of a random selection. Fig. 3 shows the prototypes returned by the Kcentres procedure. Fig. 4 shows the result of the Modeseek procedure using the neighbourhood parameter 8. Fig. 5 shows the results of the FeatSel method on the same data. Fig. 6 shows the EdiCon results with 9 prototypes found by the algorithm. The EdiCon is the only procedure that takes into account the class labels in this examples. The 20 resulting prototypes from a run of the Center Prototype Selector and Border Prototype Selector are shown in Fig. 7 and Fig. 8.

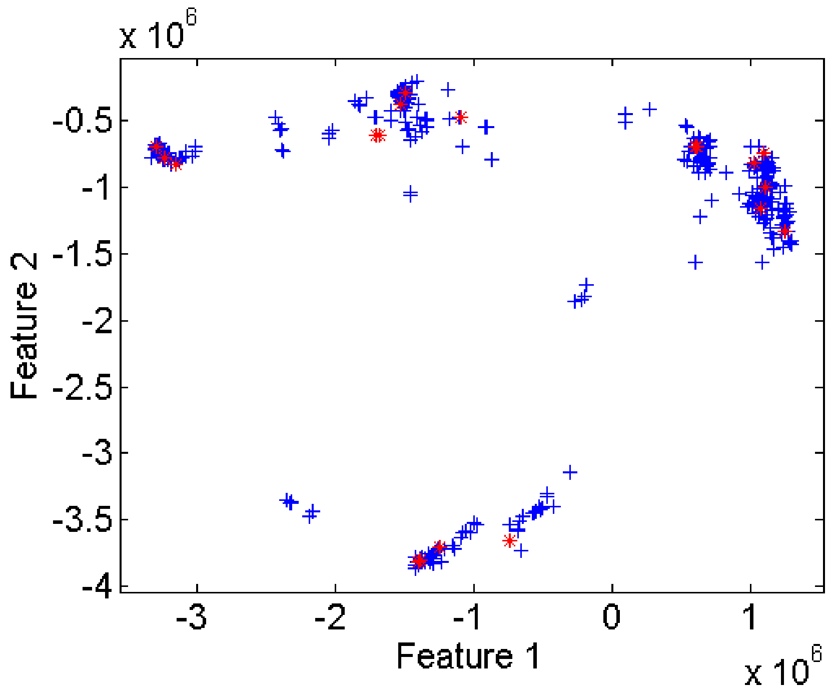


Figure 2. Random Selection

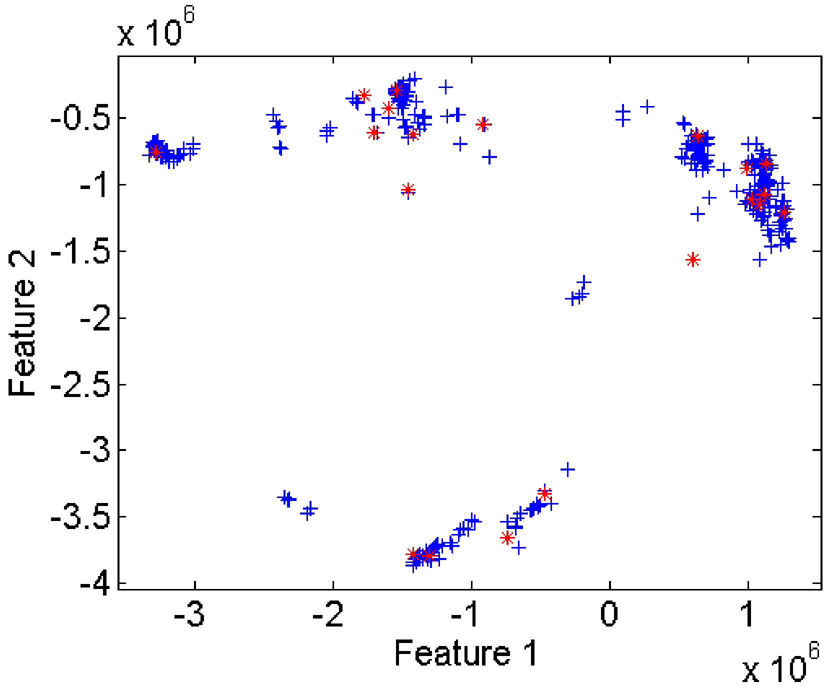


Figure 3. Kcentres

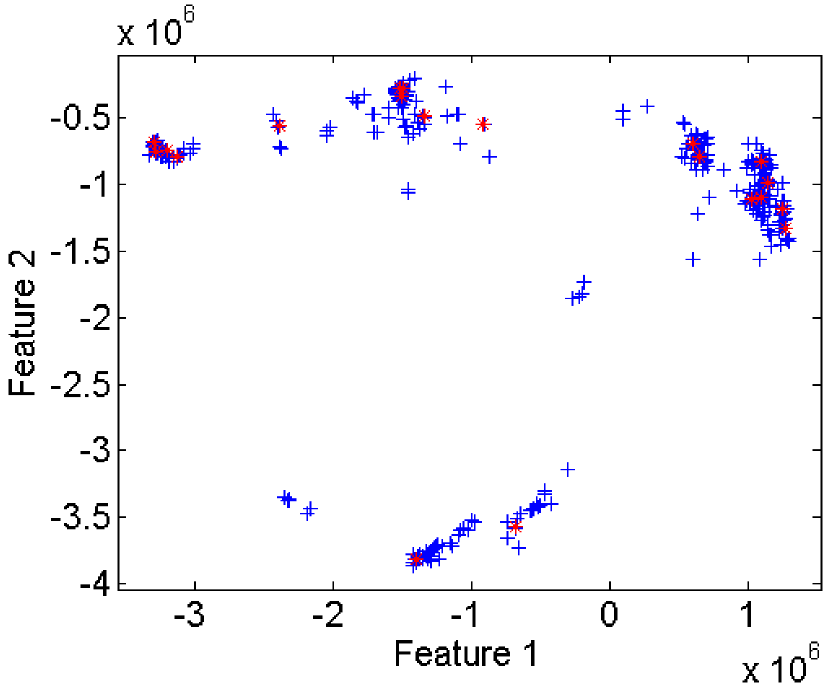


Figure 4. ModeSeek

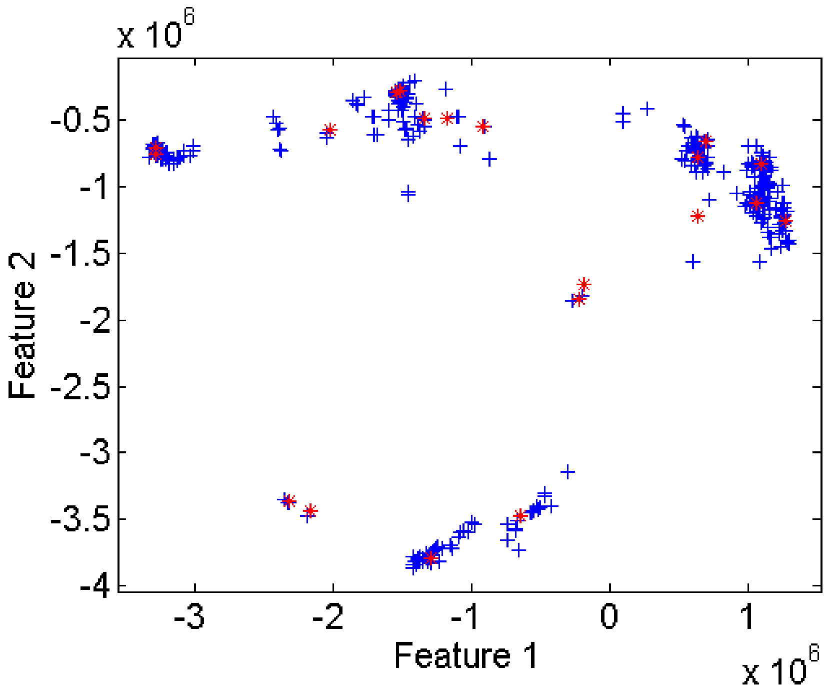


Figure 5. FeatSel

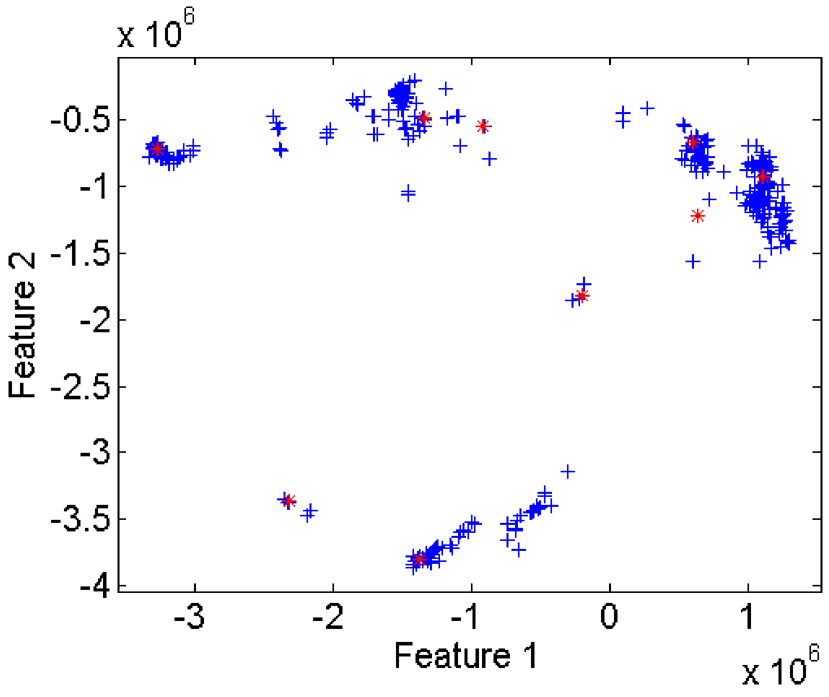


Figure 6. EdiCon

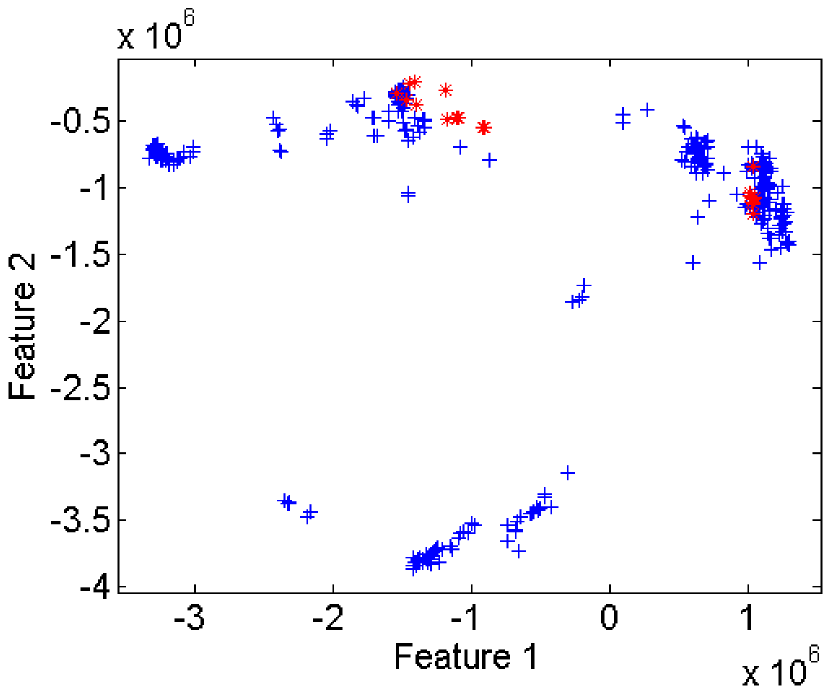


Figure 7. Center Prototype Selector

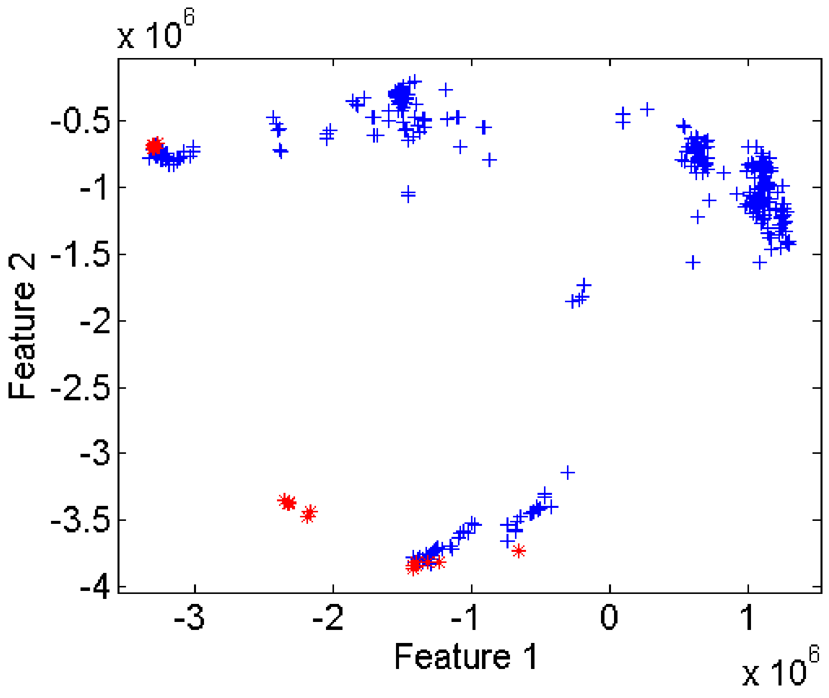


Figure 8. Border Prototype Selector

References

1. Bunke, H., Riesen, K.: Graph classication based on dissimilarity space embedding. In: N. da Vitoria Lobo et al., editor, SSSPR, LNCS 5342. (2008) 996–1008
2. Pekalska, E., Duin, R.P.W.: The Dissimilarity Representation for Pattern Recognition: Foundations And Applications (Machine Perception and Artificial Intelligence). World Scientific Publishing Co., Inc., River Edge, NJ, USA (2005)
3. Lozano, M., Sotoca, J.M., Sánchez, J.S., Pla, F., Pekalska, E., Duin, R.P.W.: Experimental study on prototype optimisation algorithms for prototype-based classification in vector spaces. Pattern Recogn. **39**(10) (2006) 1827–1838
4. Pekalska, E., Duin, R.P.W., Paclík, P.: Prototype selection for dissimilarity-based classifiers. Pattern Recogn. **39**(2) (2006) 189–208
5. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification. Wiley-Interscience Publication (2000)
6. Kohonen, T., Hynninen, J., Kangas, J., Laaksonen, J., Torkkola, K.: LVQ_PAK: The Learning Vector Quantization program package. Report A30, Helsinki University of Technology, Laboratory of Computer and Information Science (January 1996)
7. Devijver, P., Kittler, J.: Pattern Recognition: A Statistical Approach. Prentice Hall (1982)
8. Wilson, D.R., Martinez, T.R.: Reduction techniques for instance-based learning algorithms. In: Machine Learning. (2000) 257–286
9. Riesen, K., Neuhaus, M., Bunke, H.: Graph embedding in vector spaces by means of prototype selection. In Escolano, F., Vento, M., eds.: GbRPR. Volume 4538 of Lecture Notes in Computer Science., Springer (2007) 383–393
10. Spillmann, B., Neuhaus, M., Bunke, H., Pekalska, E., Duin, R.P.W.: Transforming strings to vector spaces using prototype selection. In Yeung, D.Y., Kwok, J.T., Fred, A.L.N., Roli, F., de Ridder, D., eds.: SSPR/SPR. Volume 4109 of Lecture Notes in Computer Science., Springer (2006) 287–296
11. Orozco-Alzate, M., Duin, R.P.W., Castellanos-Domínguez, G.: A generalization of dissimilarity representations using feature lines and feature planes. Pattern Recogn. Lett. **30**(3) (2009) 242–254
12. Kim, S.W.: On using a dissimilarity representation method to solve the small sample size problem for face recognition. In: ACIVS. (2006) 1174–1185
13. Orozco-Alzate, M.: Generalized Dissimilarity Representations for Pattern Recognition. PhD thesis, Universidad Nacional de Colombia Sede Manizales (October 2008)

RT_027, marzo 2010

Aprobado por el Consejo Científico CENATAV

Derechos Reservados © CENATAV 2010

Editor: Lic. Lucía González Bayona

Diseño de Portada: DCG Matilde Galindo Sánchez

RNPS No. 2142

ISSN 2072-6287

Indicaciones para los Autores:

Seguir la plantilla que aparece en www.cenatav.co.cu

C E N A T A V

7ma. No. 21812 e/218 y 222, Rpto. Siboney, Playa;

Ciudad de La Habana. Cuba. C.P. 12200

Impreso en Cuba