Prototype Selection for Dissimilarity Representation by a Genetic Algorithm

Yenisel Plasencia-Calaña*†, Edel García-Reyes*, Mauricio Orozco-Alzate‡, Robert P.W. Duin†
*Advanced Technologies Application Center, CENATAV, Havana, Cuba
Email: yplasencia,egarcia@cenatav.co.cu
†Pattern Recognition Laboratory, Delft University of Technology, The Netherlands
Email: r.duin@ieee.org
‡Departamento de Informática y Computación,
Universidad Nacional de Colombia Sede Manizales, Colombia
Email: morozcoa@bt.unal.edu.co

***Abstract*—Dissimilarities can be a powerful way to represent objects like strings, graphs and images for which it is difficult to find good features. The resulting dissimilarity space may be used to train any classifier appropriate for feature spaces. There is, however, a strong need for dimension reduction. Straightforward procedures for prototype selection as well as feature selection have been used for this in the past. Complicated sets of objects may need more advanced procedures to overcome local minima. In this paper it is shown that genetic algorithms, previously used for feature selection, may be used for building good dissimilarity spaces as well, especially when small sets of prototypes are needed for computational reasons.**

***Keywords*-dissimilarity representation; prototype selection; genetic algorithm; dimensionality reduction;**

I. Introduction

Data preparation and representation are usually required steps before an automatic analysis of the data is performed. From the pattern recognition point of view it is possible to have supervised (generalization) as well as unsupervised analyses on the basis of some representation of the patterns. Widely extended is the use of vector spaces to represent patterns, where statistical classifiers can be used and are theoretically justified by the metric or Euclidean properties assumed beforehand. More complex representation have also been studied, e.g. structural representations [1], where patterns are encoded in graphs, strings or grammars. Another way to encode patterns is by dissimilarities derived from pairwise comparisons between objects. It is possible to compute dissimilarities between vectors or structures [2], [3], and also from original data without intermediate representations (e.g. dissimilarities derived from a matching process of images directly).

If we are dealing with dissimilarities instead of distances, where an exact metric embedding is not possible, the alternatives for the classification of this data are [4]: classification by the k Nearest Neighbour (k-NN) rule, classification in dissimilarity spaces and classification after embedding the data in pseudo-Euclidean spaces. The computation of dissimilarities can be computationally demanding as in case of comparisons of images and graphs, then a reduction of the number of dissimilarities to be measured is of interest. One way to do this is by prototype selection. In dissimilarity spaces prototype selection aims at finding small sets of objects in order to decrease the dimension of the dissimilarity vectors but for the generalization step all the training objects are included. In the case of the k-NN rule and the pseudo-Euclidean spaces the prototype selection will determine the number of training samples.

For dissimilarity representations some prototype selection techniques have been investigated showing good results [5], specially when used with linear and quadratic classifiers. In [5] various techniques were compared such as Kcentres, mode seeking, forward feature selection, linear programming, editing-condensing, and a mixture of Kcentres with linear programming. It showed that forward selection is one of the best, especially for small sets of prototypes. Other prototype selection methods have been proposed in the graph and string domain [2], [3]. The methods tackle the question of how to select a small representation set for constructing the dissimilarity space. In this paper it is of our interest to study a selective scheme and not a creative scheme since one goal is to work with a given dissimilarity matrix that could have been computed directly from the initial data, and an intermediate space for creating artificial prototypes may not exist.

Genetic algorithms (GAs) have been used for feature selection [6] as well as for prototype selection [7]. In these cases, each specimen is represented by a binary chromosome whose genes are associated either to features or prototypes. However, in almost all studies where GAs have been used for prototype selection, the aim is reducing computational demands of the nearest neighbor classifier as conceived for feature spaces but not for choosing appropriate prototypes in order to build a dissimilarity space.

Fitness functions employed in the above-mentioned studies made possible to emphasize either on the maximization of the classification accuracy or on penalizing the cardinality of the representation set. Kuncheva and Jain [8] compare the GA-based selection against the sequential forward feature selection (SFS) method as well as against two classic rules

1051-4651/10 $26.00 © 2010 IEEE
DOI 10.1109/ICPR.2010.52

of condensing [9]. In general, these studies conclude that the use of GAs for feature/prototype selection is suggested in spite of its non-optimality; nonetheless, others such as [10] point out that simpler feature selection methods should be preferred because GA-based results might not be always as good as expected.

In this paper, we will show that GA-based selection can overcome local minima thus outperform the sequential forward selection of prototypes for dissimilarity space classification. Both procedures are compared on the basis of the same selection criterion, the optimization of class separability by the minimization of the leave one out (LOO) 1-NN error in the training set.

II. Prototype selection by a genetic algorithm for dissimilarity spaces

The dissimilarity space was proposed by Pekalska et al. [4]. It was postulated as a Euclidean vector space. For its construction a representation set $R = \{r_1, r_2, ..., r_k\}$ is needed, where objects belonging to this set are chosen adequately based on some criterion. Let X be the training set, R may or may not overlap with X. Once we have R, the dissimilarities of objects in X to the objects in R are computed. Any object is now represented by a vector of dissimilarities d_x to the objects in R as shown in (1).

$$d_x = [d(x, r_1)d(x, r_2)...d(x, r_k)]. \tag{1}$$

The dissimilarity space is defined by the set R. Each set of coordinates of a point in that space corresponds to the dissimilarities to the prototypes and the dimension of the space is determined by the number of prototypes selected. A good selection of R allows one to decrease the computational complexity at the cost of slightly increased error rates. Finding good criteria and algorithms for prototype selection is still an open issue. One of the best methods in [5], forward selection, used the minimization of the LOO 1-NN error as criterion. The computation of this criterion is fast as it relies on the given distances. No other computation than minimum operations are needed.

GAs are inspired in natural systems where species evolve by selection, reproduction, and mutation, ensuring better organisms as well as diversity. They are classified as global search heuristics. One drawback of the methods is that some parameters should be set by the user. Another drawback is that solutions are sensitive to initialization, and a satisfactory solution may or may not have been reached when the method stops.

In the GA vocabulary, a gene is a part of a candidate solution, a chromosome is a string of genes that represents a candidate solution, and a population is the available set of chromosomes to be explored. An important component of a GA is the fitness function that evaluates the candidate chromosomes, the GA minimizes or maximizes the value returned by this function and the best candidate is selected in each evolution cycle according to its fitness. The simplest way to encode the candidate solutions is by strings over the alphabet $\sum = \{0, 1\}$. For example a set of strings of size two over this alphabet is $\sum^2 = \{00, 01, 10, 11\}$.

The GA for prototype selection has as input parameters the dissimilarity dataset and the desired number of prototypes k. Solutions are encoded by binary strings. The first chromosome is chosen as the best, and we iterate in an evolution cycle while the fitness value of the new best chromosome improves in a new iteration. The solutions or chromosomes will evolve by the reproduction, mutation and selection operations.

The heuristic search strategy of the GA tries to avoid the local minima by introducing the random mutations. The forward search from the R initial candidates for prototypes, evaluates first each individual prototype and the one that leads to the best classification performance is kept, then the second prototype that combined with the first leads to the best classification performance is added and so on, this way of finding the solutions can cause the convergence to local minima.

III. Experiments

In order to evaluate our hypothesis on the benefits of the GA search strategy over the forward search for the problem of prototype selection for dissimilarity space classification we conducted some experiments on different dissimilarity datasets. The methods are compared in terms of classification errors of the Linear Discriminant Analysis (LDA) in the dissimilarity space while varying the number of prototypes. We compared the GA and the forward selection of prototypes using as selection criterion the LOO 1-NN error. The parameters for the GA were: 50 chromosomes for the initial population, 0.5 of mutation and reproduction probability, and 3 generations without change as stopping criterion. As a baseline we also computed the results for the random selection and for the 1-NN classifier applied directly on the total dissimilarity matrix and not in the dissimilarity space. Learning curves are used to illustrate the changing rate of learning for the LDA classifier while varying the number of prototypes.

Four different dissimilarity datasets were used for the experiments, the CatCortex, Chickenpieces-20-60, CoilYork and CoilDelftDiff datasets [11]. All of them are multiclass problems. The datasets were split randomly into training and test sets taking approximately 50 percent of the objects in each set. A comparison of different properties of the datasets can be found in Table I.

The smallest dataset is the CatCortex [12], the 65x65 dissimilarity matrix is measured in an ordinal scale and describes the connection strengths among 65 cortical areas of a cat from four regions (classes): auditory (10 samples), frontolimbic (19), somatosensory (18) and visual (18).

The dissimilarity dataset Chickenpieces-20-60 is computed from the chickenpieces image dataset [13]. From these images the edges are extracted and approximated by segments of length 20, and a string representation of the angles between the segments is derived. The dissimilarity matrix is composed by edit distances between these strings. The cost function between the angles is defined as the difference in case of substitution and as 60 in case of insertion or deletion.

The CoilYork dataset is composed by dissimilarities between graphs derived from four objects of the COIL database, the graphs are the Delaunay triangulations derived from corner points of the images [14]. The dissimilarity matrix is constructed by graph matching, using the algorithm of [15].

The CoilDelftDiff dissimilarity dataset is also computed from a set of graphs derived from four objects of the COIL database. The graphs are the Delaunay triangulations derived from corner points of the images [14]. Graphs are compared in the eigenspace with a dimensionality determined by the smallest graph in every pairwise comparison by the JoEig approach [16].

Average classification errors over 40 repetitions of the 1-NN in the dissimilarity matrix and LDA in the dissimilarity space for the random, forward and GA selection as a function of the number of prototypes, are presented in Figs. 1, 2, 3, and 4.

Table I
PROPERTIES OF DATASETS.

Datasets	# classes	# objects per class	symmetric	metric
CatCortex	4	10,19,18,18	yes	no
ChickenPieces-20-60	5	117,76,96,61,96	no	no
CoilYork	4	4x72	no	no
CoilDelftDiff	4	4x72	yes	no

It was observed that in a number of cases for small sets of prototypes the GA search can find better solutions than the forward search as it can be seen in the CoilYork, CatCortex and CoilDelftDiff datasets. But in other cases as in the ChickenPieces-20-60 dataset the GA cannot improve the solution found by the forward selection. In the CatCortex and CoilDelftDiff with prototype selection it is possible to reach or improve the 1-NN classification results at a lower computational cost since the 1-NN needs to measure all the dissimilarities to the training objects, and with prototype selection only the dissimilarities to the prototypes are measured. In the other datasets the 1-NN has a smaller error but at the cost of a superior computational demand.

IV. Discussion and Conclusion

The problem of prototype selection for building dissimilarity spaces is related to that of feature selection for feature spaces. There are, however, a few significant

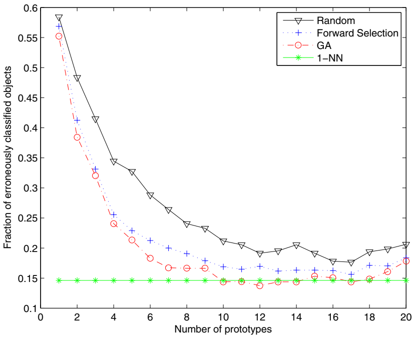

Figure 1. Classification results of LDA in the dissimilarity space and 1-NN on the CatCortex dataset.

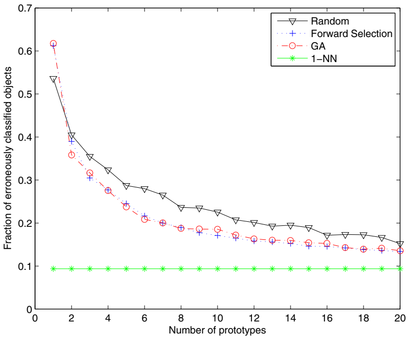

Figure 2. Classification results of LDA in the dissimilarity space and 1-NN on Chickenpieces-20-60 dataset.

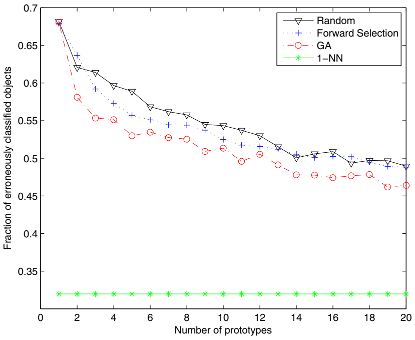

Figure 3. Classification results of LDA in the dissimilarity space and 1-NN on the CoilYork dataset.

differences. In contrast to feature sets, the set of candidate prototypes (the training set of objects) constitutes a homogeneous field: neighboring object have similar properties as prototypes. There is not really a scaling issue because all dissimilarities are in the same range. As given dissimilarities may result from clustered sets of objects and are often non-Euclidean, the search for good sets of prototypes may be hampered by local minima. On

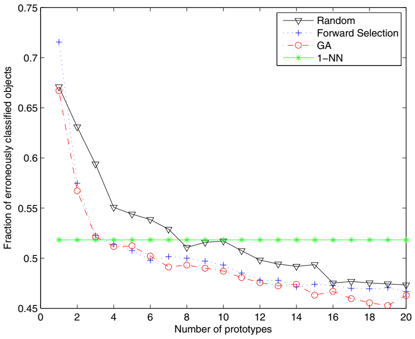

Figure 4. Classification results of LDA in the dissimilarity space and 1-NN on the CoilDelftDiff dataset.

the positive side it should be mentioned that the given dissimilarities may be used for a fast computation of separability criteria. This makes the use of GAs desirable as well as feasible.

In this paper we have shown by a set of experiments on given dissimilarity matrices that GAs may be used successfully to construct good dissimilarity spaces. There is not always a need to do this. Forward search procedures as well as even random selection may also do well. For some problems, of which we have given clear examples, GAs perform better, especially when small sets of prototypes (e.g. one to twenty) are needed and also when training sets are small and complicated. It is of importance when fast classifiers have to be built in dissimilarity spaces maximizing the classification accuracy. For larger training and prototype sets, GAs might need a careful selection of their parameter values.

Acknowledgment

We acknowledge financial support from the FET programme within the EU FP7, under the SIMBAD project (contract 213250) as well as the project ”Cálculo científico para caracterización e identificación en problemas dinámicos” (code Hermes-10722) granted by Universidad Nacional de Colombia.

References

[1] H. Bunke and K. Riesen, “Graph classication based on dissimilarity space embedding,” in *N. da Vitoria Lobo et al., editor, SSSPR, LNCS 5342*, 2008, pp. 996–1008.

[2] K. Riesen, M. Neuhaus, and H. Bunke, “Graph embedding in vector spaces by means of prototype selection.” in *GbRPR*, ser. Lecture Notes in Computer Science, F. Escolano and M. Vento, Eds., vol. 4538. Springer, 2007, pp. 383–393.

[3] B. Spillmann, M. Neuhaus, H. Bunke, E. Pekalska, and R. P. W. Duin, “Transforming strings to vector spaces using prototype selection.” in *SSPR/SPR*, ser. Lecture Notes in Computer Science, D.-Y. Yeung, J. T. Kwok, A. L. N. Fred, F. Roli, and D. de Ridder, Eds., vol. 4109. Springer, 2006, pp. 287–296.

[4] E. Pekalska and R. P. W. Duin, *The Dissimilarity Representation for Pattern Recognition: Foundations and Applications (Machine Perception and Artificial Intelligence)*. River Edge, NJ, USA: World Scientific Publishing Co., Inc., 2005.

[5] E. Pekalska, R. P. W. Duin, and P. Paclík, “Prototype selection for dissimilarity-based classifiers,” *Pattern Recogn.*, vol. 39, no. 2, pp. 189–208, 2006.

[6] W. Siedlecki and J. Sklansky, “A note on genetic algorithms for large scale feature selection,” *Pattern Recognition Letters*, vol. 10, pp. 335–347, 1989.

[7] D. A. Muresan, “Genetic algorithms for nearest neighbor,” California Institute of Technology, CA, Tech. Rep., 1997. [Online]. Available: http://www.cs.caltech.edu/~muresan/GANN/report.html

[8] L. I. Kuncheva and L. C. Jain, “Nearest neighbor classifier: Simultaneous editing and feature selection,” *Pattern Recognition Letters*, vol. 20, pp. 1149–1156, 1999.

[9] P. E. Hart, “The condensed nearest neighbor rule,” *IEEE Transactions on Information Theory*, vol. IT-14, no. 3, pp. 515–516, May 1968.

[10] A. K. Jain and D. Zongker, “Feature selection: Evaluation, application, and small sample performance,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 2, pp. 153–158, 1997.

[11] E. Pekalska and R. P. Duin, “Datasets and tools for dissimilarity analysis in pattern recognition, simbad technical report,” 2009.

[12] J. W. Scannell, C. Blakemore, and M. P. Young, “Analysis of connectivity in the cat cerebral cortex.” *Journal of Neuroscience*, vol. 15, p. 1463, 1995.

[13] H. Bunke and U. Buhler, “Applications of approximate string matching to 2d shape recognition,” *Pattern Recognition*, vol. 26, no. 12, pp. 1797–1812, December 1993. [Online]. Available: http://dx.doi.org/10.1016/0031-3203(93)90177-X

[14] B. Xiao and E. R. Hancock, “Geometric characterisation of graphs,” in *ICIAP*, 2005, pp. 471–478.

[15] S. Gold and A. Rangarajan, “A graduated assignment algorithm for graph matching,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 4, pp. 377–388, 1996.

[16] W.-J. Lee and R. P. Duin, “An inexact graph comparison approach in joint eigenspace,” in *SSPR & SPR ’08: Proceedings of the 2008 Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition*. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 35–44.