A Multiscale Approach in Combining Classifiers in Dissimilarity Representations

A. Ibba, R. P. W. Duin

ICT Group, Delft University of Technology
The Netherlands
a.ibba@tudelft.nl, r.duin@ieee.org

Keywords: multiscale, combining classifiers, dissimilarity representations, prototype selection

Abstract

The representations of real world objects based on distances (dissimilarities) has proven to be more suitable than the classic feature-based ones for many pattern recognition problems.

Measuring objects to obtain features is indeed needed to represent them, but that process can be costly, therefore selecting a reduced set of features (prototypes in dissimilarity representations) can lead to cheaper and faster solution of real life problems. The way distances are computed (namely the metric used) produces different representations of the same objects, therefore as in many cases we (or experts) cannot decide which approach is better, a combination of either dissimilarity representations or classifiers built on each of them separately can be useful . Although dissimilarity representations can be already seen as a form of classifier combination (a combination of NN classifiers), in this paper we wanted to investigate further this approach. The key point of classifier combination lies either in a proper averaging over different experts/sources or in a beneficial integration of different and possibly complementary approaches. Our aim in this paper is to investigate on the possibility to combine different dissimilarity representations of the same data, namely distances computed segmenting images using various resolutions (scales). And therefore using these multiple representations applying some prototype selection techniques to build different simple classifiers (on the obtained representation sets), and to employ then different classifier combination approaches.

1 Introduction

A representation of data different from a feature based description is based on pair-wise comparisons of objects namely distances or dissimilarities. In many cases, distances are obtained directly from raw or pre-processed measurements. There are many arguments to choose dissimilarities in particular when feature representations cannot be helpful in discriminating different classes of objects, or in case the experts are not able to define proper features, or the data lies in high-dimensional spaces (too many features/measurements). But also the intrinsic nature of the problem at hand is quite relevant: for instance measures of curves and shapes are good examples of cases in which a dissimilarity representation might perform better in the recognition tasks.

We have chosen a dissimilarity based approach instead of a kernel based one (similarities) because it is not therefore needed to deal only with squared matrices, but indeed we look for reduced representations. Dissimilarity representations also allows the use of huge data, after a proper dimensionality reduction obtained via prototype selection techniques, another advantage is the possibility to use a larger variety of different classifiers rather than only SVM ones. And the constraint of fulfilling the Mercer's theorem is not a problem in this approach, therefore allowing for the use of non-euclidean and even non-metric distances that in many are able to better describe the data at hand which can lie on non-euclidean manifolds.

We chose to combine different approaches based on dissimilarities because experiments in Multiple Classifier Systems as well as life experience show that a proper fusion of complementary expertise leads to a better understanding of the problem and, usually, to reach better solutions.

In our paper we have selected a finite number of representations based on different scales of the same data and proposed three different methods for combining them, obtaining a single representation with

two approaches (using distances computed on the basis of the same metric but with different parameters):

- formulating a function depending on these representations (namely a weighted sum).
- concatenating (extending) the dissimilarity representations and applying different prototype selection approaches [1].
- combining different classifiers built on the various representations of the given data.

With respect to the extended dissimilarity representation and to the combination of classifiers built on different representations it is important to underline the importance of finding effective ways to allign the prototype selection along the different matrices composing it, in order to minimize both the costs involved in measurement and the computational complexity that can easily grow whether different sets (with respect to the the sub-matrices) are used instead of employing a consistent selection approach.

2 Data description

For our experiments we have chosen to use dissimilarity matrices from the H.Bunke's "Chicken Pieces Silhouttes Database".

2.1 Chicken Pieces Silhouttes Database

The chicken pieces dataset contains several silhouettes of chicken pieces. The contour line of each silhouette is extracted using an edge detector. Then the resulting contour line is approximated by a sequence of normalized vectors of constant length. A string consisting of the angles between consecutive vectors is constructed from this vector sequence , which leads to a rotation-invariant cyclic string of relative angles representing the original chicken piece silhouette. An illustration of the processing steps can be found in figure 1. The distances between these strings have been computed using an efficient cyclic string edit distance algorithm to make them rotation-invariant edit distances.

This dataset consists of 446 images of chicken pieces. Each piece belongs to one of five categories, which represent specific parts of the chicken: wing (117 samples), back (76), drumstick (96), thigh and back (61), and breast (96). Each one of the given image is in binary format containing the silhouette of a particular piece. Pieces were placed in a natural way without considering orientation. In figure 1 we can see an example image of the wing class (a), edge detection (b), and the edges approximated by straight line segments of fixed length (c). Figure 2 shows the results for segment lengths of 7, 10, 15 and 20 pixels. The applied normalization value *n* indicates that the contour has been normalized to segments of *n*-pixels length. These segments could have been chosen as symbols for the strings. But due to the following two constraints:

- the figures have to be rotation invariant,
- there should be a mirror symmetry

better string representation has been used.
Therefore, the sequence of angles between the segments were chosen as the string representation. Additionally, the approximate algorithm of Bunke and Buhler [2], which handles rotation invariance and axis symmetry, was applied. Cost Functions: The cost functions are defined as the angle difference in case of substitution and as a constant k in case of inserting or deleting a symbol. In the following equation, α and β are arbitrary angles, and ε stands for the empty symbol. $c_k(\alpha \rightarrow \beta) = |\alpha\text{-}\beta|$ (angle difference)
$c_k(\varepsilon \rightarrow \alpha) = k$
$c_k(\alpha \rightarrow \varepsilon) = k$

In this paper we used segment lenghts of $\{20; 25; 29; 30; 31\}$ with the value k = 45.

3 Experimental setup

A dissimilarity representation of objects is based on pairwise comparisons and is expressed e.g. as a $N \times N$ dissimilarity matrix ***D(T,T)***, where each element corresponds to a dissimilarity between a pair of objects in the dataset T (as defined in the previous sections). Hence each object *x* is represented by a vector of proximities ***D(x,T)*** to the objects in T. A new example *z*, represented by ***D(z,T)***, is classified to a specific class if it is sufficiently similar to one or more objects within the class.
Assume a representation set $R := [p_1, p_2, ..., p_n]$ as a collection of n prototype objects and a dissimilarity measure *d*, computed or derived from the objects directly, their sensor representations, or some other initial representation. To maintain generality, a notation of $d(x, z)$ is used when objects *x* and *z* are quantitatively compared (namely a distance d between them is computed), *d* is required to be nonnegative and to obey the reflexivity condition, $d(x, x) = 0$, but it might be non-metric. An object *x* is represented as a vector of the dissimilarities computed between x and the prototypes from R, i.e. $D(x, R) = [d(x, p_1), d(x, p_2), ..., d(x, p_n)]$.
For a set *T* of *N* objects, it extends to an $N \times n$ dissimilarity matrix $D(T, R)$, which is a dissimilarity representation we want to learn from.

The selection of a representation set for the construction of classifiers in a dissimilarity space serves a similar goal as the selection of prototypes to be used

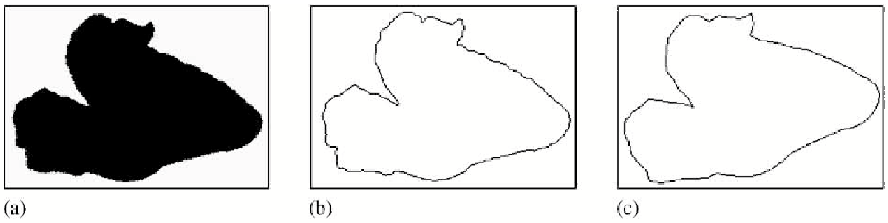

Figure 1: *Chicken pieces dataset: (a) silhouette image; (b) extracted contour line; and (c) normalized string*

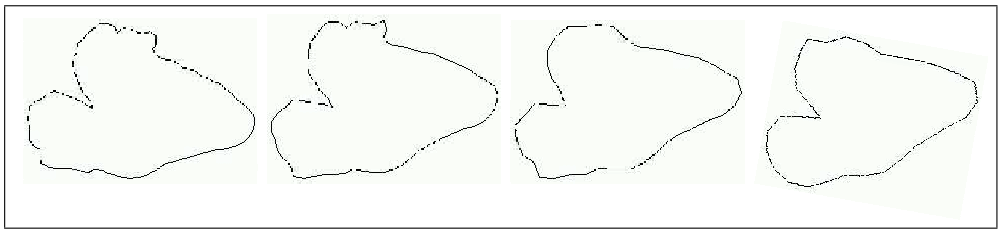

Figure 2: *Segment lengths of 7, 10, 15, 20 pixels*

by the NN rule: the minimization of the set of dissimilarities to be measured for the classification of new incoming objects.

In the experiments the distance matrices have been handled as classic dataset having been divided into two equal part and then used as design set a *L* and a testing set *S*. The experiments have been performed twenty times on randomly chosen training and testing sets for each diastance matrix under analysis.
In each single experiment, the dataset was randomly split into two equal-sized sets: the design set *L* and the testing set *S*, *L* serves for obtaining both the representation set *R* and the training set *T*. The size of representation set has been varied from 1% to about 70% of the design set with 15 steps in total. After *R* is chosen, the training set *T* has been defined as a submatrix of the design set made by the distances of all the objects (present in design set) to the selected prototypes. Each time a 2-fold cross-validation has been applied by switching the role of the two sets (*T* and *S*).

As it has been previously said in the introduction, in this paper we have chosen to use different combining approaches based on dissimilarity matrices, namely:

1. A weighted sum of different dissimilarity representations.

   $$D(T,R)_{sum} = \Sigma_{i=1}^{k} D^{(i)}(T,R)_{max}$$

   where $D^{(i)}(T,R)_{max} = \alpha_i D^{(i)}(T,R)$.
   The weights α_i are computed in such a way that the maximum distances are equal over all the matrices. This scaling procedure has been applied to avoid that the combining method used might be biased by representations with larger distances.

2. Another approach used in this work is based on a combination of the separate dissimilarity representations into a bigger one containing all of them. The result of this concatenation is called an *extended* representation: $D_{ext}(T,R) = [D^{(1)}(T,R), D^{(2)}(T,R), ..., D^{(k)}(T,R)]$, as a result this representation is a *kr*-dimensions space where *r* is the size of the representation set R, therefore the strategies used to select these prototype sets play a key role and they will be described later.

3. In our experiments we have also used a more classical combining approach employing three combining rules (*mean*, *max*, *product*) to normal density based classifiers trained on each distinct representation $D^{(i)}(T,R)$.

In our combining experiments we have used the four representations ($\{20; 25; 30; 31\}$ pixel lenghts)*around* the "29" norm (with α = 45 which is the cost function leading to the best classification performances).

In all these three different combining methods explained above we used a prototype selection technique called *k-centres* described as follows:

Assume c classes: $\omega_1, ..., \omega_c$. Let *T* be a training set and let T_{ω_i} describe the training objects of the class ω_i. This technique is applied to each class separately. For each class ω_i, it tries to choose *k* objects

such that they are evenly distributed with respect to the dissimilarity information $D(T_{\omega_i}, T_{\omega_i})$. The algorithm proceeds as follows:

1. Select an initial set $R_{\omega_i} := \{p_1^{(i)}, p_2^{(i)}, ..., p_k^{(i)}\}$ consisting of k objects, e.g. randomly chosen, from T_{ω_i}.

2. For each $x \in T_{\omega_i}$ find its nearest neighbor in R_{ω_i}. Let $J_j, j = 1, 2, ..., k$, be a subset of T_{ω_i} consisting of objects that yield the same nearest neighbor $p_j^{(i)}$ in R_{ω_i}. This means that $T_{\omega_i} = \cup_{j=1}^{k} J_j$.

3. For each J_j find its center c_j, that is the object for which the maximum distance to all other objects in J_j is minimum (this value is called the radius of J_j).

4. For each center c_j, if $c_j \neq p_j^{(i)}$, then replace $p_j^{(i)}$ by c_j in R_{ω_i}. If any replacement is done, then return to (2), otherwise the procedure **ends**. The final representation set *R* consists of all sets R_{ω_i}.

This technique has been applied to the three combining methods explained above with a varying size of the given prototype sets. In the case of the *extended* this procedure cannot be used on the entire matrix (since it's not squared and therefore not symmetric), therefore we employed the *k-centres* selection technique on each single $D^{(i)}(T, T)$, replicating each k set of prototypes along each $D^{(i)}(T, T)$ namely alligning then training and testing (on an indipendent test set S) the used classifiers (LDC, QDC) on the whole $D^{(i)}(T, R)$. Then we have computed a set of prototypes (for each fixed size of T) in a different way for each single combining method. Although our aim is not to focus our research on the the prototype selection techniques we have compared our results using the set of prototypes determined on the weighted matrix and on each distinct matrix in the combining approach leading to a total of four different strategies applied on both the datasets at hand. To study the differences of these approaches in particular between the more classical one and the first two ones, we have plotted the curves of the classification errors versus the varying size of the representation set. The classifiers used in this work are the linear and quadratic normal density based classifiers. After all computations the average classification error (for each single configuration) and its related standard deviation have been computed over the 20 independent runs.

4 Results

We have therefore presented our results in the form of curves showing the averaged classification error (over a two-fold crossvalidation repeated 20 times).

In figure: 3 it is possible to compare the linear density based classifier performances as a function of the varying size of the representation sets for a total of four cases. The first four plots norm-$\{20; 25; 30; 31\}$ are related to the "Chicken pieces silhouettes databases" dataset with the cited norms (and k=45). These represent the closest four representations of the data *around* the best perfoming norm: the *29*-pixels segmentation case (namely the "Best" curve in the plot). In this figure (3) we have also plotted the averaged classification errors for the "weighted sum" and the "extended" dissimilarity representation constructed concatenating the four non-optimal ones and *alligning* the set of prototypes (for each one of the four matrices) to the one obtained for the "weighted sum" approach, therefore it has been obviously rescaled (original version in fig: 5) to make it comparable to the other experiments. Moreover we have produced the plot related to the performance of the combination ("mean ldc comb")of normal density based classifiers trained using the first four representations with a consistent protype selection (namely alligned along all the four matrices).

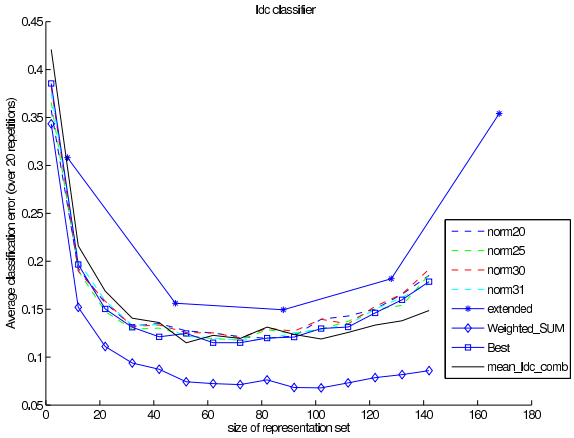

Figure 3: *Average classification error for the linear density based classifier over a varying size of the representation set*

All the studied methods approach an error value close to their overall minimum for a number of prototypes nearly equal to 50. It is pretty clear that the weighted sum approach outperforms all the other studied combining methods and in particular it is interesting to notice that the related averaged classification error is significantly lower than its components' one and than the optimal one (29-pixels), also the combining approach ("mean ldc comb") leads better performances than this 29-pixels representation although in this case these improvements are not as clear as for the sum approach, while the "extended" representation is not giving any improvement with respect to its component. In figure: 4 are shown similar plots as in the previous one, with the main difference that the quadratic normal density based is used instead

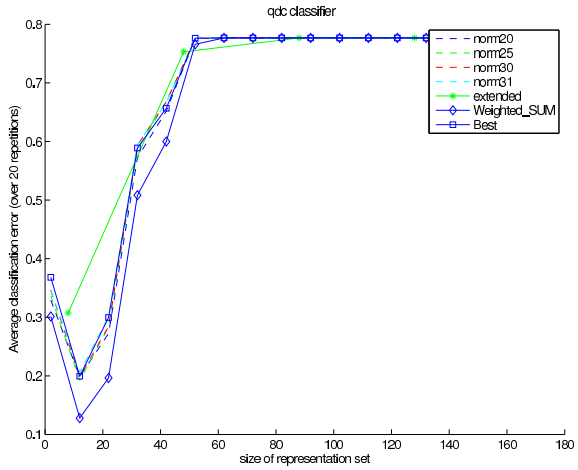

Figure 4: *Average classification error for the quadratic density based classifier for the over a varying size of the representation set.*

of the linear one. Since the dataset at hand is composed of objects belonging to five different classes (different chicken pieces) it is quite obvious that the classification performances start to deteriorate much faster (for smaller number of prototypes) than for the linear case. Also for this classifier (quadratic) the "weighted sum" method is prevailing over the other studied approaches.

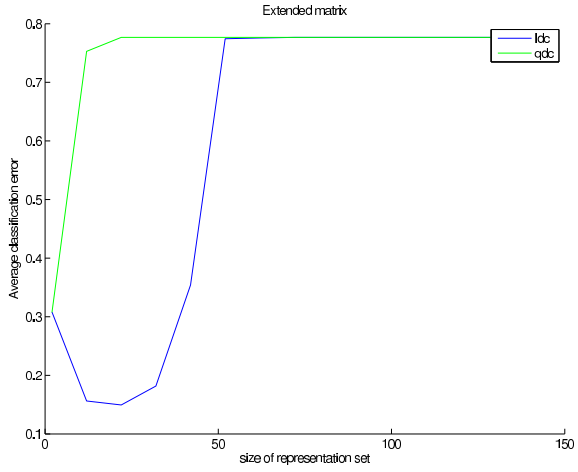

Figure 5: *Average classification error (over 20 runs) for the "extended" representation of the linear and quadratic density based classifier as a function of the representation set's size.*

The figure: 5 shows the plots related to the "extended" representation using linear and quadratic classifiers, in this case we are obviously interested on the very first part of the plot that has been plotted rescaled respectively in figures: 3 and 4. The averaged errors of the combination of linear classifier built on each single representation using three different combining rules (mean, product, max) are presented in figure: 6. The *mean* results have been plotted in figure: 3 to show that this method can perform slightly better than each individual representation composing the combination.

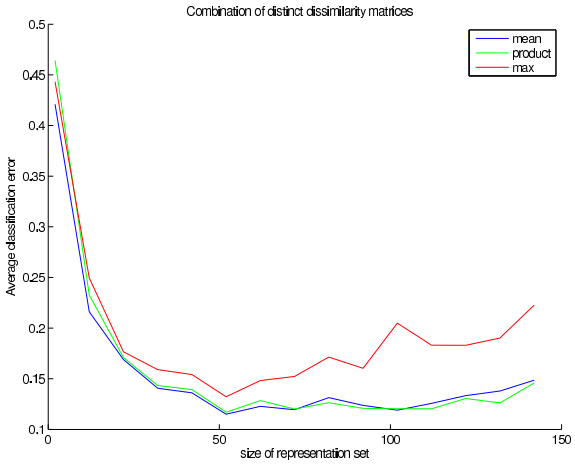

Figure 6: *Average classification error (over 20 runs) for the combined normal linear classifiers built on each distinct matrix as a function of the representation set's size, using three different combination rules.*

5 Conclusions

In this paper we have studied three different ways of combining either representations or classifiers built on those distinct representations. Our focus has been put on investigating whether one of those methods could lead to performance improvements while reducing the number of selected prototypes. We have studied the possibility of alligning the same set of prototypes along all the given dissimilarity representations in order to further minimize both the measurement (as these can be taken at once) and the computational effort. Our results have shown a drammatic improvement of the classification accuracy using the simplest approach studied and also the mean linear combiner based on the representations *alligned* on the "weighted sum" selected prototypes is leading to some (although much smaller) improvements. Therefore could be of interest to further analyze the reasons of this phenomena, whether it is peculiar of this dataset and whether this behaviour could change with respect of the prototype selection technique employed.

References

[1] E. Pękalska and R.P.W. Duin. On combining dissimilarity representations. *MCS 2001, LNCS 2096*, pages 359–368, 2001.

[2] H. Bunke and U. Bühler. Applications of approximate string matching to 2d shape recognition. *Pattern Recognition*, 26(12):1797–1812, 1993.

[3] C. Lai, D.M.J. Tax, R.P.W. Duin, E. Pękalska, and P. Paclík. A study on combining image

representations for image classification and retrieval. *International Journal of Pattern Recognition and Artificial Intelligence*, 18(5):867–890, 2004.

[4] E. Pękalska and R. P. W. Duin. Dissimilarity representations allow for building good classifiers. *Pattern Recognition Letters*, 23(8):943–956, 2002.

[5] E. Pękalska, R. P. W. Duin, and P. Paclík. Prototype selection for dissimilarity based classifiers. *Pattern Recognition*, 39(2):189–208, 2006.

[6] E. Pękalska. *Dissimilarity representations in pattern recognition*. PhD thesis, Delft University of Technology, 2005.

[7] R.P.W. Duin. The combining classifier: To train or not to train? In *International Conference on Pattern Recognition*, volume II, pages 765–770, Quebec City, Canada, 2002.

[8] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. John Wiley & Sons, Inc., 2nd edition, 2001.

[9] Andrew R. Webb. *Statistical Pattern Recognition 2nd. edition*. John Wiley & Sons Ltd., 2002.

[10] L. Kuncheva. *Combining pattern classifiers*. Wiley Inter-Science, 2004.

[11] K. Fukunaga. *Introduction to statistical pattern recognition*. Academic Press, 1990.