A New Classifier Combination Scheme Using Clustering Ensemble

Miguel A. Duval-Poo, Joan Sosa-García, Alejandro Guerra-Gandón, Sandro Vega-Pons, and José Ruiz-Shulcloper

Advanced Technologies Application Center (CENATAV), Havana, Cuba
{mduval,jsosa,aguerra,svega,jshulcloper}@cenatav.co.cu

Abstract. Combination of multiple classifiers has been shown to increase classification accuracy in many application domains. Besides, the use of cluster analysis techniques in supervised classification tasks has shown that they can enhance the quality of the classification results. This is based on the fact that clusters can provide supplementary constraints that may improve the generalization capability of the classifiers. In this paper we introduce a new classifier combination scheme which is based on the Decision Templates Combiner. The proposed scheme uses the same concept of representing the classifiers decision as a vector in an intermediate feature space and builds more representatives decision templates by using clustering ensembles. An experimental evaluation was carried out on several synthetic and real datasets. The results show that the proposed scheme increases the classification accuracy over the Decision Templates Combiner, and other classical classifier combinations methods.

Keywords: Classifier Combination, Decision Templates, Clustering Ensemble.

1 Introduction

There are several areas in pattern recognition where the use of reliable and accurate classifiers is necessary. Traditionally, these problems have been solved with the use of a single classifier. However, one single classifier cannot always reach the desired classification accuracy for a specific problem. One way to improve the results of a single classifier is by combining multiple base classifiers [1].

On the other hand, the idea of combining different clustering results (clustering ensemble) emerged as an alternative approach to improve the quality of clustering algorithms [2]. This is possible because the combination process can compensate possible errors in individual clustering results.

Recently, new methods for combining supervised classifiers that use cluster analysis as a tool for improving the classification results have been presented [3, 4, 5]. Some even combine both ensembles of classifiers and clusterers [6]. The main motivations for doing such combinations is that the use of unsupervised models can provide a variety of supplementary constraints for classifying new data [6].

L. Alvarez et al. (Eds.): CIARP 2012, LNCS 7441, pp. 154–161, 2012.

Classifiers output can be categorized into three levels: abstract level (class label), rank level (rank order of class labels), and measurement level (soft labels) [7]. Depending on the form of the information delivered by base classifiers, different schemes has been proposed for combining multiple classifiers decisions. The simplest and more frequently considered rule to combine class labels is the Majority Vote. In this scheme, base classifiers output is used as class votes and the most voted class is returned. There are other class labels combiners like Bayesian Combination and Behavior Knowledge Space [1]. On the other hand, soft labels can be combined by using simple rules like the sum, product and average of the support values given by the base classifiers to each class [8]. Others schemes like Fuzzy Integral [8] and Dempster−Shafer [9] can be also used for combining soft labels. Another way to combine soft classifiers results is to see them as features in an intermediate feature space. At this point any classifier, also called meta-classifier, can be applied to make the final class decision. Following this approach, one of the most widely used method is the *Decision Templates Combiner* (DTC) [10]. DTC is a robust scheme that builds class templates in the intermediate feature space by using the true class labels of the objects in a training set. Then, a new object is classified by comparing the base classifiers output for this object, to each class template.

In this paper, we introduce a new classifier combination scheme that uses clustering ensemble tools for finding more representative templates for a class. To do that, we combine partitions obtained by two different procedures. First, a set of partitions is generated by grouping objects taking into account their proximity values in the intermediate feature space. In the second case, another partition is obtained by using the information of the true class labels in the training set. Finally, all partitions are combined to obtain a consensus one, where each cluster centroid can be viewed as a new decision template.

The remainder of this paper is organized as follows: In Section 2 the DTC is described. The proposed combination scheme is introduced in Section 3. Section 4 presents an experimental study performed on several datasets, in which the proposed scheme is compared to DTC and other classifier combination methods. Finally, conclusions and future works are presented in Section 5.

2 Decision Templates Combiner

Let us view the problem of classifying an object $\mathbf{x}$ into c classes using L individual classifiers, where $\mathbf{x}$ is a tuple of some n-dimensional space $\mathbb{F}^n$. Let $\Omega = \{\omega_1, \omega_2, \ldots, \omega_c\}$ be the set of class labels and $\mathbb{D} = \{D_1, D_2, \ldots, D_L\}$ be the ensemble of supervised classifiers. Each classifier D_i is a function $D_i : \mathbb{F}^n \to \mathbb{R}^c$ that returns a c-dimensional vector $[d_{i,1}(\mathbf{x}), d_{i,2}(\mathbf{x}), \ldots, d_{i,c}(\mathbf{x})]$ where $d_{i,j}(\mathbf{x})$ denote the support that classifier D_i gives to the hypothesis that $\mathbf{x}$ belongs to the class ω_j. In addition, let us assume that a labeled data set $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N\}$, $\mathbf{z}_i \in \mathbb{F}^n$ is available, which is used to train the classifier combination scheme: both the individual classifiers and the combiner.

Kuncheva [1] proposed to organize the L classifiers output for a particular input $\mathbf{x}$ as a matrix called *decision profile* ($DP(\mathbf{x})$).

$$DP(\mathbf{x}) = \begin{bmatrix} d_{1,1}(\mathbf{x}) \ \ldots \ d_{1,j}(\mathbf{x}) \ \ldots \ d_{1,c}(\mathbf{x}) \\ d_{i,1}(\mathbf{x}) \ \ldots \ d_{i,j}(\mathbf{x}) \ \ldots \ d_{i,c}(\mathbf{x}) \\ d_{L,1}(\mathbf{x}) \ \ldots \ d_{L,j}(\mathbf{x}) \ \ldots \ d_{L,c}(\mathbf{x}) \end{bmatrix} \tag{1}$$

One way of using the $DP(\mathbf{x})$ matrix for combining classifiers is to treat the values $d_{i,j}(\mathbf{x})$ as features in a new space called *intermediate feature space*. Then, another supervised classifier returns the class label taking as input the data in this space. One of the most widely used method that follows this approach is the Decision Templates Combiner (DTC).

The idea behind the DTC is to remember the most typical decision profile for each class ω_j, which is called *decision template* (DT_j). The decision template DT_j for class ω_j is the average of the decision profiles of the elements of the training set $\mathbf{Z}$, labeled in class ω_j:

$$DT_j = \frac{1}{N_j} \sum_{\substack{\mathbf{z}_k \in \mathbf{Z} \\ l(\mathbf{z}_k)=\omega_j}} DP(\mathbf{z}_k) \tag{2}$$

where N_j is the number of elements in $\mathbf{Z}$ that belong to the class ω_j and $l(\mathbf{z})$ represents the class label of $\mathbf{z}$. After constructing the DTs matrices in the training phase, when a new object $\mathbf{x}$ is submitted for classification, the DTC scheme matches $DP(\mathbf{x})$ to DT_j, $j = 1, \ldots, c$ and produces soft class labels by:

$$\mu_j(\mathbf{x}) = S(DP(\mathbf{x}), DT_j), \quad j = 1, \ldots, c \tag{3}$$

where S is a similarity measure. The higher the similarity between the $DP(\mathbf{x})$ and the DT_j, the higher the support for the class ω_j.

3 Classifier Combination Using Clustering Ensemble

As we previously said, this method is based on the idea of representing objects with the support values given by the base classifiers for each class. In other words, each object $\mathbf{x}$ can be represented in the $(L \cdot c)$ - dimensional intermediate feature space as a vector $dv(\mathbf{x})$ obtained by concatenating its $DP(\mathbf{x})$ rows. This way the training set $\mathbf{Z}$ is mapped into a new set $\mathbf{Y}$ in the intermediate feature space in the following way $\mathbf{Y} = \{\mathbf{y}_i \in \mathbb{R}^{L \cdot c} \mid \mathbf{y}_i = dv(\mathbf{z}_i), i = 1, \ldots, N\}$.

In this space, the original DTs in (2) can be viewed as the centroids of the clusters in the *ground-truth* partition. The *ground-truth* partition of $\mathbf{Y}$ is defined as $P^{gt} = \{G_1^{gt}, G_2^{gt}, \ldots, G_c^{gt}\}$, where each cluster $G_j^{gt} = \{\mathbf{y}_k \in \mathbf{Y} \mid l(\mathbf{y}_k) = \omega_j\}$. DTC represents in a single DT the most typical decision behavior for each class. However, in some cases, there could be more than one typical behavior for a group of objects belonging to the same class. In other words, there could be cluster centroids in P^{gt} with a low representative power. In these cases, representing a class by a single DT can lead to a not representative template for this class.

Therefore, to build more representative DTs we propose to use cluster analysis tools. In particular, we use clustering ensemble to combine partitions in which objects are grouped by they decision behavior with the *ground-truth* partition. As result, a new *consensus partition* is build where each cluster centroid can be view as a more representative DT. Each one of these DTs, instead of being associated to a single class label, gives a support value to each class.

Formally, a set $\mathbb{P}^{db} = \{P_1, P_2, \ldots, P_M\}$ of partitions of $\mathbf{Y}$ is generated by using different clustering algorithms or the same algorithm with different parameter initialization. In each $P_i = \{G_1^i, G_2^i, \ldots, G_{q_i}^i\}$, G_j^i is the j^{th} cluster of the i^{th} partition, for all $i = 1, \ldots, M$. Next, a partition ensemble $\mathbb{P}$ is build by joining the partition set $\mathbb{P}^{db}$ with the *ground-truth* partition, i.e. $\mathbb{P} = \mathbb{P}^{db} \cup \{P^{gt}\}$. Then, the *consensus partition* $P^* = \{G_1^*, G_2^*, \ldots, G_{q_*}^*\}$ is built as

$$P^* = \arg\max_{P \in \mathbb{P}_{\mathbf{Y}}} \sum_{i=1}^{M+1} b_i \Gamma(P, P_i) \tag{4}$$

where $\mathbb{P}_{\mathbf{Y}}$ is the set of all possible partitions with the set of objects $\mathbf{Y}$, and Γ is a similarity measure between partitions. Each b_i is a weight associated to the partition P_i. In particular, P_{M+1} represents the *ground-truth* partition P^{gt} and b_{M+1} is its associated weight. The influence of $\mathbb{P}^{db}$ and P^{gt} in the combination process can be handled by using the b_i weights.

Finally, for each cluster G_k^* in the *consensus partition*, a centroid e_k is calculated and used as a DT. Besides, a class support vector $[\mu_1^k, \mu_2^k, \ldots, \mu_c^k]$ is computed. Each μ_j^k denotes the support given by the cluster centroid e_k to the fact that an object belongs to class ω_j. This support is defined as:

$$\mu_j^k = \frac{|\{\mathbf{y} \in G_k^* \mid l(\mathbf{y}) = \omega_j\}|}{|G_k^*|} \tag{5}$$

where the numerator represents the number of elements of the cluster G_k^* that belong to class ω_j and the denominator is the total number of elements in the cluster G_k^*.

Once the model is trained, we obtain a collection of q_* centroids and support vectors. When a new object $\mathbf{x}$ is wanted to be classified, the similarity between its representative vector $dv(\mathbf{x})$ in the intermediate feature space and all the e_k centroids is computed. Finally, the proposed scheme returns the support values of the most similar centroid to $\mathbf{x}$:

$$\mu_j(\mathbf{x}) = \mu_j^r, \quad r = \arg\max_k S(DP(\mathbf{x}), e_k) \tag{6}$$

Notice that the final *consensus partition* does not have to necessary possess c clusters. Therefore, a class can be represented by more than one DT. In addition, the DTC can be viewed as a particular case of the proposed scheme when the partition ensemble only contains the *ground-truth* partition, $\mathbb{P} = \{P^{gt}\}$. In this case, the *consensus partition* will be in fact the same *ground-truth* partition and its clusters centroid will be the original DTs. Although the class supports will not be the same that the originally calculated by the DTC (3), notice that both

will return the same class label, wherever the class with maximum support value is selected as crisp label.

4 Experimental Evaluation

Experiments with six numerical datasets were conducted to evaluate the proposed scheme. Three datasets were selected from the UCI Machine Learning Repository [11] (Iris, Wine, SPECT) while the other three are 2D synthetic datasets (Half-Rings, Cassini, Smiley), see Fig. 1. A description of the datasets is presented in Table 1.

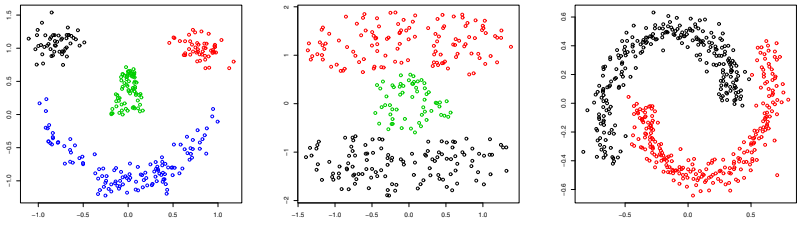

Fig. 1. 2D synthetic datasets. Smiley (Left), Cassini (Center), Half-Rings (Right).

Table 1. Description of the datasets

Dataset	No. Instances	No. Classes	No. Attributes	Instances per classes
Smiley	200	4	2	33-33-50-84
Cassini	300	3	2	120-60-120
Half-Rings	200	2	2	100-100
Iris	150	3	4	50-50-50
Wine	178	3	13	59-71-48
SPECT	267	2	22	55-212

Ten *fast decision tree learners* (REPTree), implemented in Weka [12], were used as the classifier ensemble. Each classifier in the ensemble was configured with their default parameters and with a different random seed. The criterion used for measuring the results was the classification accuracy.

In all experiments, each dataset was divided in 3 groups of equal size in which objects were randomly assigned. The first group was used to train the classifier ensemble. In this case, each base classifier was trained with 70% of the objects in the group, randomly selected with replacement. The second group was used for training the combination schemes and the third for testing. This process was repeated 100 times and the final accuracy was calculated as an average over the 100 iterations.

For the proposed method, an alpha parameter, $\alpha \in [0, 1]$, was used for assigning the partition weights. The weight used for the *ground-truth* partition

was $b_{gt} = 10(1-\alpha)$. Besides, $M = \lfloor 10\alpha \rfloor$ partitions were generated by using k-means with euclidian distance and a random number of clusters k between c and $3c$. Each one of these partitions was assigned with a weight $b_i = 1,\ i = 1, \ldots, M$. The α parameter is used to simultaneously control the weight of the *ground-truth* partition and the number of partitions in $\mathbb{P}^{db}$. Notice that the *ground-truth* partition is more taken into account in the combination process as the α value decreases. On the contrary, higher α values increase the influence of $\mathbb{P}^{db}$ partitions in the combination process.

In order to find the *consensus partition*, the following procedure [13] is used. First, a co-association matrix is build, where each position (i, j) of this matrix has a value that represents how many times the objects x_i and x_j are in the same cluster for all partitions in $\mathbb{P}$. This co-association matrix is used as a similarity measure between objects. Then, the *consensus partition* is obtained by applying a hierarchial clustering algorithm. In this case, we use the Group Average algorithm and the *highest lifetime* criterion to select the most representative level in the hierarchy.

The similarity measure S used in our experiments is defined based on the euclidian distance in the following way:

$$S(x, y) = 1 - \frac{1}{L \cdot c}\|x - y\|_2 \tag{7}$$

Two experiments were carried out. In the first, the accuracy was evaluated with different values of the α parameter, see Fig. 2. The effect of the $\mathbb{P}^{db}$ partitions on the accuracy can be analyzed by the application of different α values.

In the second experiment (Table 2), the proposed scheme (CCCE) was tested on each dataset and then compared with the classifier ensemble average (CEA), the best (BBC) and worst (WBC) base classifier of the ensemble. Also was compared with other combination methods like: Majority Vote (MV), the average combination function (AVG), the Dempster−Shafer Combination (D-F) and the

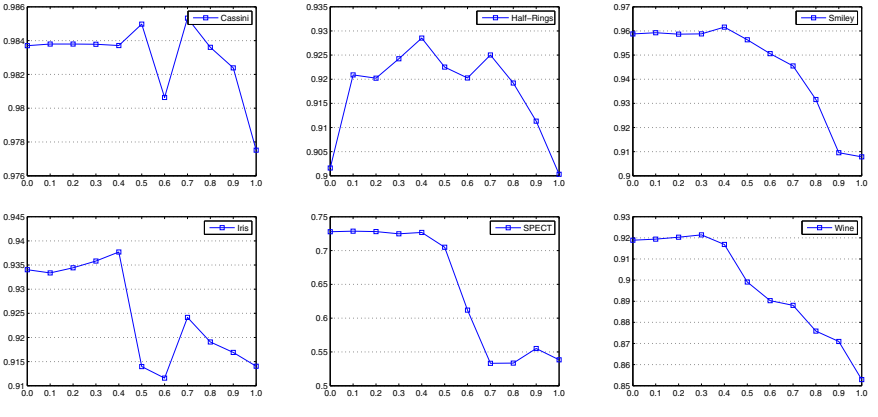

Fig. 2. Accuracy of DTCE on the datasets with different values of the α parameter

Table 2. Accuracy (%) for the different methods over the six datasets

	Synthetic			Real		
Method	Smiley	Cassini	Half-Rings	Iris	Wine	SPECT
WBC	89.863	97.022	84.157	86.324	77.377	57.191
CEA	91.716	97.249	85.826	88.127	78.933	59.864
BBC	92.427	97.752	86.908	90.648	80.016	61.419
MV	95.303	98.197	87.917	92.901	90.259	58.135
AVG	95.627	98.212	88.852	92.923	90.474	57.135
D-F	95.901	98.280	88.803	93.242	90.835	72.413
DTC	95.876	98.370	90.162	93.404	91.892	72.485
CCCE	**96.160**	**98.533**	**92.852**	**93.770**	**92.142**	**72.865**

Decision Template Combiner (DTC). For the DTC, the similarity measure in (7) was used. Besides, for the proposed CCCE, the α parameter employed in each dataset was the one for which the highest accuracy was reached in the first experiment.

4.1 Results

Notice that in the first experiment, when $\alpha = 0$, CCCE is equivalent to the DTC. However, as the α parameter is increased, the partitions in $\mathbb{P}^{db}$ will have more influence in the determination of the consensus partition. As Fig. 2 shows, the use of those partitions can improve the accuracy. However, when the partitions in $\mathbb{P}^{db}$ are more taken into account in the determination of the consensus partition, the accuracy drastically decreases. That is why it is very important to establish a correct balance between the number of partitions in $\mathbb{P}^{db}$ and the *ground-truth*, by adjusting the partitions weights.

In the second experiment, the results in Table 2 show that the proposed CCCE outperforms the accuracy of the base classifiers in the ensemble and the other combination methods, particularly the DTC. This shows that using the information of the $\mathbb{P}^{db}$ partitions, instead of only using the *ground-truth* partition like in the DTC, more representative DTs can be build. This way, the accuracy can be increased.

5 Conclusions

In this paper, we have proposed a new scheme to combine multiple classifiers by using clustering ensemble. It uses the main idea of the DTC of building class templates in an intermediate feature space. The use of unsupervised learning tools, specially clustering ensemble, helps to build more representative class templates in the intermediate feature space. An experimental comparison was carried out with other classifier combination methods, including the DTC, on several datasets. The results show that the proposed scheme improves the accuracy of the classifier ensemble and the other combination methods. This supports

the idea that ensembles of supervised and not supervised classifiers can complement to each other and produce high quality classification results. As future work, we will perform a more exhaustive experimental evaluation using more complex datasets. Additionally, we will evaluate different clustering algorithms, consensus functions, the weights b_i associated with each partition as well as new distance measures.

References

[1] Kuncheva, L.I.: Combining Pattern Classifiers. Methods and Algorithms. John Wiley & Sons, New York (2004)

[2] Vega-Pons, S., Ruiz-Shulcloper, J.: A survey of clustering ensemble algorithms. International Journal of Pattern Recognition and Artificial Intelligence 25(3), 337–372 (2011)

[3] Jurek, A., Bi, Y., Wu, S., Nugent, C.: Classification by Cluster Analysis: A New Meta-Learning Based Approach. In: Sansone, C., Kittler, J., Roli, F. (eds.) MCS 2011. LNCS, vol. 6713, pp. 259–268. Springer, Heidelberg (2011)

[4] Gao, J., Liangy, F., Fanz, W., Sun, Y., Han, J.: Graph-based consensus maximization among multiple supervised and unsupervised models. In: 23rd Annual Conference on Neural Information Processing Systems, pp. 1–9 (2009)

[5] Ma, X., Luo, P., Zhuang, F., He, Q., Shi, Z., Shen, Z.: Combining supervised and unsupervised models via unconstrained probabilistic embedding. In: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (2011)

[6] Acharya, A., Hruschka, E.R., Ghosh, J., Acharyya, S.: $\mathbf{C}^3$E: A Framework for Combining Ensembles of Classifiers and Clusterers. In: Sansone, C., Kittler, J., Roli, F. (eds.) MCS 2011. LNCS, vol. 6713, pp. 269–278. Springer, Heidelberg (2011)

[7] Xu, L., Krzyzak, A., Suen, C.: Methods of combining multiple classifiers and their applications to handwriting recognition. IEEE Transactions on Systems, Man and Cybernetics 22(3), 418–435 (1992)

[8] Kuncheva, L.: Combining classifiers: Soft computing solutions, pp. 427–452. World Scientific (2001)

[9] Rogova, G.: Combining the results of several neural network classifiers. Neural Networks 7(5), 777–781 (1994)

[10] Kuncheva, L., Bezdek, J., Duin, R.: Decision templates for multiple classifier fusion: an experimental comparison. Pattern Recognition 34(2), 299–314 (2001)

[11] Frank, A., Asuncion, A.: UCI machine learning repository (2010), `http://archive.ics.uci.edu/ml`

[12] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: An update. SIGKDD Explorations 11(1) (2009)

[13] Fred, A., Jain, A.: Combining multiple clustering using evidence accumulation. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(6), 835–850 (2005)