Dynamic and Static Weighting in Classifier Fusion

Rosa M. Valdovinos[1], J. Salvador Sánchez[2], and Ricardo Barandela[1]

[1] Instituto Tecnológico de Toluca, Av. Tecnológico s/n, 52140 Metepec, México
{li_rmvr,rbarandela}@hotmail.com
[2] Dept. Llenguatges i Sistemes Informàtics, Universitat Jaume I, 12071 Castelló, Spain
sanchez@uji.es

Abstract. When a Multiple Classifier System is employed, one of the most popular methods to accomplish the classifier fusion is the simple majority voting. However, when the performance of the ensemble members is not uniform, the efficiency of this type of voting is affected negatively. In this paper, a comparison between simple and weighted voting (both dynamic and static) is presented. New weighting methods, mainly in the direction of the dynamic approach, are also introduced. Experimental results with several real-problem data sets demonstrate the advantages of the weighting strategies over the simple voting scheme. When comparing the dynamic and the static approaches, results show that the dynamic weighting is superior to the static strategy in terms of classification accuracy.

1 Introduction

A multiple classifier system (MCS) is a set of individual classifiers whose decisions are combined when classifying new patterns. There are many different reasons for combining multiple classifiers to solve a given learning problem [6], [12]. First, MCSs try to exploit the local different behavior of the individual classifiers to improve the accuracy of the overall system. Second, in some cases MCS might not be better than the single best classifier but can diminish or eliminate the risk of picking an inadequate single classifier. Another reason for using MCS arises from the limited representational capability of learning algorithms. It is possible that the classifier space considered for the problem does not contain the optimal classifier.

Let $D = \{ D_1, ..., D_h \}$ be a set of classifiers. Each classifier assigns an input feature vector $\mathbf{x} \in \Re^n$ to one of the c problem classes. The output of a MCS is an h-dimensional vector containing the decisions of each of the h individual classifiers:

$$[D_1(\mathbf{x}),..., D_h(\mathbf{x})]^T \quad (1)$$

It is accepted that there are two main strategies in combining classifiers: selection and fusion. In classifier selection, each individual classifier is supposed to be an expert in a part of the feature space and therefore, we select only one classifier to label the input vector **x**. In classifier fusion, each component is supposed to have knowledge of the whole feature space and correspondingly, all individual classifiers decide the label of the input vector.

Focusing on the fusion strategy, the combination can be made in many different ways. The simplest one employs the majority rule in a plain voting system [4]. More

J.S. Marques et al. (Eds.): IbPRIA 2005, LNCS 3523, pp. 59–66, 2005.

elaborated schemes use weighted voting rules, in which each individual component is associated with a different weight [5]. The final decision can be made by majority, average [6], minority, medium [7], product of votes, or using some other more complex methods [8], [9], [10], [19].

In the present work, some methods for weighting the individual components in a MCS are proposed, and their effectiveness is empirically tested over real data sets. Three of these methods correspond to the so-called dynamic weighting, by using the distances to a pattern. The last method, which belongs to the static weighting strategy, estimates the leaving-one-out error produced by each classifier in order to set the weights of each component [21].

From now on, the rest of the paper is organized as follows. Sect. 2 provides a brief review of the main issues related to classifier fusion and makes a very simple categorization of weighting methods, distinguishing between dynamic and static weighting of classifiers. Moreover, seveal weighting procedures are also introduced in Sect. 2. The experimental results are discussed in Sect. 3. Finally, some conclusions and possible further extensions are given in Sect. 4.

2 Classifier Fusion

As pointed out in Sect. 1, classifier fusion assumes that all individual classifiers are competitive, instead of complementary. For this reason, each component takes part in the decision of classifying an input test pattern.

In the simple voting (by majority), the final decision is taken according to the number of votes given by the individual classifiers to each one of the classes, thus assigning the test pattern to the class that has obtained a majority of votes. When working with data sets that contain more than two classes, in the final decision ties among some classes are very frequently obtained. To solve this problem, several criteria can be considered. For instance, to randomly take the decision, or to implement an additional classifier whose ultimate goal is to bias the decision toward a certain class [15].

An important issue that has strongly called the attention of many researchers is the error rate associated to the simple voting method and to the individual components of a MCS. Hansen and Salomon [17] show that if each one of the classifiers being combined has an error rate less than 50%, it may be expected that the accuracy of the ensemble improve when more components are added to the system. However, this assumption not always is fulfilled. In this context, Matan [18] asserts that in some cases, the simple voting might perform even worse than any of the members of the MCS. Thus some weighting method can be employed in order to partially overcome these difficulties.

A weighted voting method has the potential to make the MCS more robust to the choice of the number of individual classifiers. Two general approaches to weighting can be remarked: dynamic weighting and static weighting of classifiers. In the dynamic strategy, the weights assigned to the individual classifiers can change for each test pattern. On the contrary, in the static weighting, the weights are computed for each classifier in the training phase, and they are maintained constant during the classification of the test patterns.

In the following sections, several weighting functions, both from the dynamic and the static categories, are explored. It has to be noted that in the present work, all the individual classifiers correspond to the 1-NN (Nearest Neighbor) rule [16]. This is a well-known supervised non-parametric classifier that combines conceptual and implementational simplicity with an asymptotic error rate conveniently bounded in terms of the optimal Bayes error. In its classical manifestation, given a set of *m* previously labeled instances (or training set, TS), this classifier assigns any input test pattern to the class indicated by the label of the closest example in the TS. The extension of this rule corresponds to the *k*-NN classifier, which consists of assigning an input pattern to the class most frequently represented among the *k* closest training instances.

2.1 Dudani's Dynamic Weighting

A weighted *k*-NN rule for classifying new patterns was first proposed by Dudani [3]. The votes of the *k* nearest neighbors are weighted by a function of their distance to the test pattern. In his original proposal, a neighbor with smaller distance is weighted more heavily than one with a greater distance: the nearest neighbor gets a weight of 1, the furthest neighbor a weight of 0, and the other weights are scaled linearly to the interval in between (Eq. 2):

$$w_j = \begin{cases} \dfrac{d_k - d_j}{d_k - d_1} & \text{if } d_k \neq d_1 \\ 1 & \text{otherwise} \end{cases} \tag{2}$$

where d_j denotes the distance of the *j*'th nearest neighbor to the test pattern, d_1 is the distance of the nearest neighbor, and d_k indicates the distance of the furthest (*k*'th) neighbor.

Now, this function will be here applied to make the dynamic weighting of the individual components in an ensemble. Correspondingly, the value of *k* (that is, the number of nearest neighbors in Dudani's rule) will be replaced by the number of classifiers *h* that constitute the MCS. The procedure to assign the weights can be described as follows:

1. Let d_j (j = 1, …, h) be the distance of an input test vector **x** to its nearest neighbor in the j'th individual classifier.
2. Sort the h distances in increasing order: d_1, …, d_h.
3. Weight classifier D_j by means of function in Eq. 2.

2.2 Dynamic Weighting by Index

Another weighting function is here considered. Like in Dudani's method, the *h* distances of the test pattern **x** to its nearest neighbor in each individual classifier have also to be sorted. In this case, each classifier D_j is weighted according to the following function:

$$w_j = h - j + 1 \tag{3}$$

where *j* represents the index of an individual classifier after sorting the corresponding *h* distances.

Consider a MCS consisting of three individual classifiers D = {D_1, D_2, D_3}. The distance of the nearest neighbor to a given test pattern **x** by means of each classifier is d_1, d_2, and d_3, respectively. Now suppose that $d_2 < d_1 < d_3$. Thus after sorting the three distances, the index of classifier D_1 is 2, the index of D_2 is 1, and the index of D_3 is 3. Consequently, by applying the weighting function in Eq. 3, the resulting weights are $w_1 = 3 - 2 + 1 = 2$, $w_2 = 3 - 1 + 1 = 3$, and $w_3 = 3 - 3 + 1 = 1$.

2.3 Dynamic Weighting by Averaged Distances

We here propose a novel weighting function, which is based on the computation of averaged distances. In summary, the aim of this new dynamic weighting procedure is to reward (by assigning the highest weight) the individual classifier with the nearest neighbor to the input test pattern. The rationale behind this is that such a classifier probably corresponds to that with the highest accuracy in the classification of the given test pattern. Thus each classifier D_j will be weighted by means of the function shown in Eq. 4:

$$w_j = \frac{\sum_{i=1}^{h} d_i}{d_j} \tag{4}$$

Note that, by using this weighting function, we effectively accomplish the goal previously stated, that is, the individual classifier with the smallest distance will get the highest weight, while the one with the greatest distance will obtain the lowest weight.

2.4 Static Weighting by Leaving-One-Out Error Estimate

While the previous methods weight the individual components of a MCS in a dynamic manner, the last proposal corresponds to the static category. In this sense, weighting will be here performed in the training phase by means of the leaving-one-out error estimate method. To this end, for each individual classifier D*j*, the following function e_j is defined:

$$e_j = \frac{1}{m} \sum_{x \in S} e(y, x) \tag{5}$$

where m denotes the number of patterns in a training sample S, x represents a training instance, y is the nearest neighbor of x in $S - \{x\}$, and $e(y, x)$ is defined as follows:

$$e(y, x) = \begin{cases} 0 & \text{if } L(y) = L(x) \\ 1 & \text{otherwise} \end{cases} \tag{6}$$

where $L(x)$ is the class label of a pattern x, and $L(y)$ indicates the class label of a pattern y.

By using the error function just introduced, each individual classifier D_j will be weighted according to the function in Eq. 7:

$$w_j = 1 - \frac{e_j / m}{\sum_{i=1}^{h} e_i} \tag{7}$$

Note that this weight is directly related to the amount of errors produced by each individual classifier. Thus the classifier with the smallest error will be assigned the highest weight, while the one with the greatest error will obtain the lowest weight.

3 Experimental Results

The results here reported correspond to the experiments over six real data sets taken from the UCI Machine Learning Database Repository [11]. For each data set, the 5-fold cross-validation error estimate method was employed: 80% of the available patterns were for training purposes and 20% for the test set.

The integration of the MCS was performed by manipulating the patterns [12] for each of the classes, thus obtaining three different individual classifiers with four variants:

- Sequential selection [1], [2] (Sel1)
- Random selection with no replacement [1], [2] (Sel2)
- Selection with Bagging [13] (Sel3)
- Selection with Boosting [14] (Sel4)

The experimental results given in Table 1 correspond to the averages of the general accuracy in the fusion, by technique of pattern selection and method of weighting. The 1-NN classification accuracy for each entire original TS (i.e., with no combination) has also been included as the baseline classifier. Analogously, the results for the MCS with simple voting (no weighting) are reported for comparison purposes.

From results in Table 1, some preliminary conclusions can be drawn. First, for all data sets there exists at least one classifier fusion technique whose classification accuracy is higher than that obtained when using the whole TS (i.e., with no combination). Second, comparing the four selection methods, in general Sel1 and Sel4 clearly outperform the other two selection approaches (namely, random with no replacement and bagging), independent of the voting scheme adopted. On the other hand, focusing on sequential selection (Sel1) and boosting (Sel4), the accuracy of Sel1 results superior to that of Sel4 in most cases (22 out of 30).

If we now compare the simple and the weighted voting schemes, we can observe that in all data sets, we can find a weighting technique with better results than those of the simple majority voting. The Dudani's weighting outperforms all the other methods in Liver database. The weighting by index is the best in Cancer and Glass domains. The weighting by averaged distances achieves the highest accuracy in Heart, Pima and Vehicle databases.

Finally, with respect to differences in accuracy between dynamic and static weighting, it has to be especially remarked the fact that results of the static strategy are always inferior to those of the dynamic approach. As can be seen, although differences are not significant, the static weighting does not seem to present any advantage with respect to the dynamic weightings.

Table 1. Averaged accuracy of different classifier fusion methods. Values in italics indicate the best selection method for each voting scheme and each data set. Boldface is used to emphasize the highest accuracy for each problem

	Cancer	Heart	Liver	Pima	Glass	Vehicle
Original TS	95.62	58.15	65.22	65.88	70.00	64.24
Simple voting						
Sel1	*96.93*	*65.19*	*63.77*	68.89	*68.00*	*64.48*
Sel2	66.42	50.37	57.10	59.35	56.50	62.10
Sel3	72.12	45.19	50.14	60.00	60.50	60.55
Sel4	94.16	57.78	62.03	*70.07*	62.50	60.43
Dudani's weighting						
Sel1	95.62	58.15	***65.51***	*68.37*	*70.00*	*64.24*
Sel2	68.47	52.96	56.23	59.08	67.00	61.02
Sel3	74.16	47.41	52.17	60.26	65.00	60.91
Sel4	*95.89*	*58.52*	60.87	67.58	66.50	64.24
Weighting by index						
Sel1	95.91	*61.11*	*62.61*	68.24	***71.00***	*64.48*
Sel2	65.84	54.07	53.04	62.09	62.00	62.34
Sel3	72.41	47.78	49.28	60.92	61.50	60.79
Sel4	***99.27***	57.41	59.42	*70.07*	66.00	62.81
Weighting by averaged distances						
Sel1	*96.50*	***65.56***	*65.22*	68.37	*68.00*	***64.72***
Sel2	62.04	49.63	57.10	59.08	59.00	59.00
Sel3	70.80	45.93	50.14	60.26	62.50	63.41
Sel4	93.58	57.78	62.32	***70.85***	63.00	61.50
Static weighting						
Sel1	*96.93*	*65.19*	*63.77*	68.89	*68.50*	*63.65*
Sel2	66.42	50.37	57.10	59.35	56.00	62.93
Sel3	72.12	45.19	50.14	60.00	60.50	59.84
Sel4	94.16	59.63	62.03	*70.07*	63.00	61.03

4 Conclusions and Future Work

In a MCS, performance mainly depends on the accuracy of the individual classifiers and on the specific way of combining the individual decisions. Correspondingly, it results crucial to appropriately handle the combination of decisions in order to attain the most accurate system. In the present work, several weighting methods, both from the dynamic and static approaches, have been introduced and empirically compared with the simple majority voting scheme.

From the experiments carried out, our study shows that the weighting voting clearly outperforms the simple voting procedure, which erroneously assumes the uniform performance of the individual components of a MCS. Another issue to remark is that the dynamic weighting is superior to the static strategy, in terms of classification accuracy.

At this moment, it has to be admitted that it results difficult enough to propose one of the dynamic weightings as the best method. In fact, differences among them are more or less significant depending on each particular database. Nevertheless, one can

see that the weighting by averaged distances achieves the highest accuracy in 3 out of 6 problems (50% of the cases), while the weighting by index in 2 out of 6 databases (33% of the cases).

Future work is primarily addressed to investigate other weighting functions applied to classifier fusion. For instance, the inverse distance function proposed by Shepard [20] could represent a good alternative to other weighted voting schemes with low classification accuracy. On the other hand, the results reported in this paper should be viewed as a first step towards a more complete understanding of the behavior of the weighted voting procedures and consequently, it is still necessary to perform a more extensive analysis of the dynamic and static weighting strategies over a larger number of synthetic and real problems.

Acknowledgements

This work has been partially supported by the Programme Alβan, the European Union Programme of High Level Scholarships for Latin America (Scholarship No. E04D033674MX), and by grants TIC2003-08496 from the Spanish CICYT, GV04A/705 from Generalitat Valenciana, and P1-1B2004-08 from Fundació Caixa Castelló-Bancaixa.

References

1. Barandela, R., Valdovinos, R.M., Sánchez, J.S.: New applications of ensembles of classifiers, Pattern Analysis and Applications 6 (2003) 245-256.
2. Valdovinos, R.M., Barandela, R.: Sistema de Múltiples Clasificadores. Una alternativa para la Escalabilidad de Algoritmos, In: Proc. of the 9th Intl. Conference of Research on Computer Sciences, Puebla, Mexico (2002).
3. Dudani, S.A.: The distance weighted k-nearest neighbor rule, IEEE Trans. on Systems, Man and Cybernetics 6 (1976) 325-327.
4. Kuncheva, L.I., Kountchev, R.K.: Generating classifier outputs of fixed accuracy and diversity, Pattern Recognition Letters 23 (2002) 593–600.
5. Woods, K., Kegelmeyer Jr., W.P, Bowyer, K.,: Combination of multiple classifiers using local accuracy estimates, IEEE Trans. on Pattern Analysis and Machine Intelligence 19 (1997) 405-410.
6. Kuncheva, L.I.: Using measures of similarity and inclusion for multiple classifier fusion by decision templates, Fuzzy Sets and Systems 122 (2001) 401-407.
7. Chen, D., Cheng, X.: An asymptotic analysis of some expert fusion methods, Pattern Recognition Letters 22 (2001) 901–904.
8. Kuncheva, L.I., Bezdek, J.C., Duin, R.P.W.: Decision templates for multiple classifier fusion, Pattern Recognition 34 (2001) 299-314.
9. Ho, T.-K.: Complexity of classification problems and comparative advantages of combined classifiers, In: Proc. of the 1st Intl. Workshop on Multiple Classifier Systems, Springer (2000) 97-106.
10. Bahler, D., Navarro, L.: Methods for combining heterogeneous sets of classifiers, In: Proc. of the 17th Natl. Conference on Artificial Intelligence (AAAI-2000), Workshop on New Research Problems for Machine Learning (2000).
11. Merz, C.J., Murphy, P.M.: UCI Repository of Machine Learning Databases, Dept. of Information and Computer Science, Univ. of California, Irvine, CA (1998).

12. Dietterich, G.T.: Machine learning research: four current directions, AI Magazine 18 (1997) 97–136.
13. Breiman, L.: Bagging predictors, Machine Learning 24 (1996) 123-140.
14. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm, In: Proc. of the 13th Intl. Conference on Machine Learning, Morgan Kaufmann (1996) 148-156.
15. Kubat, M., Cooperson Jr., M.: Voting nearest neighbor subclassifiers, In: Proc. of the 17th Intl. Conference on Machine Learning, Morgan Kaufmann, Stanford, CA (2000) 503-510.
16. Dasaraty, B.V..: Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques, IEEE Computer Society press, Los Alamitos, CA (1991).
17. Hansen, L.K., Salomon, P. : Neural network ensembles, IEEE Trans. on Pattern Analysis and Machine Intelligence 12 (1990) 993-1001.
18. Matan, O.: On voting ensembles of classifiers, In: Proc. of the 13th Natl. Conference on Artificial Intelligence (AAAI-96), Workshop on Integrating Multiple Learned Models (1996) 84–88.
19. Ho, T.-K., Hull, J.J., Srihari, S.N.: Combination of Decisions by Multiple Classifiers, Structured Document Image Analysis, In: Springer-Verlag, Heidelberg (1992) 188–202.
20. Shepard, R.N.: Toward a universal law of generalization for psychological science, Science 237 (1987) 1317-1323.
21. Verikasa, A., Lipnickasb A., Malmqvista, K., Bacauskieneb, M., Gelzinisb, A.: Soft combination of neural classifiers: a comparative study, Pattern Recognition Letters 20 (1999) 429-444.