Brighton, UK, 2000

Clustering-and-Selection Model for Classifier Combination

Ludmila I. Kuncheva
School of Informatics, University of Wales, Bangor
Bangor, Gwynedd, LL57 1UT, United Kingdom,
(*l.i.kuncheva@bangor.ac.uk*)

Abstract

We devise a simple clustering-and-selection algorithm based on a probabilistic interpretation of classifier selection. First, the data set is clustered into K clusters and then the most successful classifier for a given cluster is nominated to label the inputs in the Voronoi cell of the cluster centroid. The proposed method is compared experimentally with minimum, maximum, product and average. Given are also the results from the Naive Bayes method, Behavior-Knowledge Space (BKS) method, the best individual and the oracle.

Keywords: Multiple classifier systems, Classifier combination, Classifier selection, Clustering and Selection.

1 Introduction

Let $\mathcal{D} = \{D_1, D_2, \ldots, D_L\}$ be a set of classifiers and $\Omega = \{\omega_1, \ldots, \omega_c\}$ be a set of class labels. Each classifier gets as its input a feature vector $\mathbf{x} \in \Re^n$ and assigns it to a class label from Ω, i.e., $D_i : \Re^n \to \Omega$. In many cases the classifier output is a c-dimensional vector with supports to the classes, i.e.,

$$D_i(\mathbf{x}) = [d_{i,1}(\mathbf{x}), \ldots, d_{i,c}(\mathbf{x})]^T. \quad (1)$$

Without loss of generality we can restrict $d_{i,j}(\mathbf{x})$ within the interval $[0, 1]$, $i = 1, \ldots, L$, $j = 1, \ldots, c$, and call the classifier outputs "soft labels". Thus, $d_{i,j}(\mathbf{x})$ is the degree of "support" given by classifier D_i to the hypothesis that $\mathbf{x}$ comes from class ω_j (most often an estimate of the posterior probability $P(\omega_i|\mathbf{x})$). Combining classifiers means to find a class label for $\mathbf{x}$ based on the L classifier outputs $D_1(\mathbf{x}), \ldots, D_L(\mathbf{x})$. Again, we can find a vector with c final degrees of support for the classes as a soft label for $\mathbf{x}$, denoted

$$D(\mathbf{x}) = [\mu_1(\mathbf{x}), \ldots, \mu_c(\mathbf{x})]^T. \quad (2)$$

If a crisp class label of $\mathbf{x}$ is needed, we can use the maximum membership rule: Assign $\mathbf{x}$ to class ω_s iff,

$$\mu_s(\mathbf{x}) \geq \mu_t(\mathbf{x}), \forall t = 1, \ldots, c. \quad (3)$$

Ties are resolved arbitrarily. The minimum-error classifiers is recovered from (3) when $\mu_i(\mathbf{x}) = P(\omega_i|\mathbf{x})$.

There are generally two types of combinations: *classifier selection* and *classifier fusion* as named in [9]. The presumption in classifier selection is that each classifier is "an expert" in some local area of the feature space. When a feature vector $\mathbf{x} \in \Re^n$ is submitted for classification, the classifier responsible for the vicinity of $\mathbf{x}$ is given the highest credit to label $\mathbf{x}$. We can nominate exactly one classifier to make the decision, as in [6], or more than one "local expert", as in [1, 4, 8].

2 A probabilistic view

Classifier selection has been proposed in the form of a composite classifier system by Dasarathy and Sheela in 1978 [2]. They combine a linear classifier and a k-nearest neighbor (k-nn) classifier. The authors suggest to identify a conflict domain in the feature space and use k-nn in that domain while using the linear classifier elsewhere. In [5], the classification procedure switches between a set of possible classifiers, based on the *certainty* of the current classification decision. This idea mimics the decision making in real life situations, e.g., in medical diagnostics, where help is sought if the confidence of the current decision-maker is not high enough. Rastrigin and Erenstein [6] propose to use the set of classifiers together with a *meta-classifier* which decides in whose region of competence the input $\mathbf{x}$ falls. The nominated classifier is then responsible for the decision. Below we show why this idea works.

Let $\Re^n$ be divided into K regions of competence, $K > 1$. Denote the regions by $R_1, \ldots, R_K$. These regions are not related to the classification regions, nor do they need to have a specific shape or size.

An example of partitioning into regions is shown in Figure 1. Depicted is a 15-point data set in $\Re^2$ with two

class labels: squares and snowflakes. The *two classification regions* defined by the nearest neighbor classifier are overlayed using Voronoi diagrams. Shaded is the classification region for class "squares". Four *selection* regions are set up, R_1, R_2, R_3, and R_4 to be used in classifier selection.

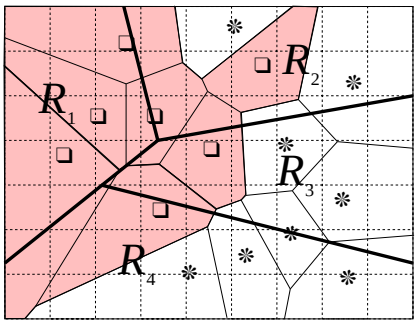

Figure 1: An example of partitioning the feature space with two classification regions into four selection regions

During training of the multiple classifier system we decide which classifier from $\mathcal{D} = \{D_1, \ldots, D_L\}$ we should nominate for each region R_j. Thus, the number of classifiers L is not necessarily equal to the number of regions K. Some classifiers might never be nominated and therefore they are not needed in the operation of the combination scheme. Curiously, even the classifier with the highest average accuracy over the whole feature space might be dropped from the final set of classifiers. On the other hand, one classifier might be nominated for more than one region.

Let D^* be the classifier with the highest average accuracy amongst the elements of $\mathcal{D}$ over the whole feature space $\Re^n$. Denote by $P(D_i|R_j)$ the probability of correct classification by D_i in region R_j. Let $D_{i(j)} \in \mathcal{D}$ be the classifier responsible for region $R_j, j = 1, \ldots, K$. The overall probability of correct classification of our classifier selection system is

$$P_c = \sum_{j=1}^{K} P(R_j) P_c(R_j) = \sum_{j=1}^{K} P(R_j) P(D_{i(j)}|R_j). \quad (4)$$

where $P(R_j)$ is the probability that an input $\mathbf{x}$ drawn from the distribution of the problem falls in R_j. To maximize P_c, we assign $D_{i(j)}$ so that

$$P(D_{i(j)}|R_j) \geq P(D_t|R_j), \forall\, t = 1, \ldots, L. \quad (5)$$

Ties are broken randomly. From (4) and (5),

$$P_c \geq \sum_{j=1}^{K} P(R_j) P(D^*|R_j) = P(D^*). \quad (6)$$

The above equation shows that the combined scheme performs better than the best classifier D^* in the pool $\mathcal{D}$, *regardless of the way the feature space has been partitioned.* The only condition (and, of course, the trickiest one) is to ensure that $D_{i(j)}$ *is* the best amongst the L classifiers in $\mathcal{D}$ for region R_j. The extent to which this is satisfied determines the success of the classifier selection model. The regions R_j can be formed by overlaying a coarse grid on the feature space, by using competitive learning, as suggested by Verikas, et al. [8] or simply by c-means clustering. The clustering-based approaches ensure that the regions have a relatively high number of points in them so that the estimates of the probabilities $P(D_i|R_j)$ can be reliable.

3 Clustering-and-Selection

We design a simple (static) classifier selection method called **clustering and selection**. Figure 2 shows the training, and Figure 3, the operation of the model.

Clustering and selection (training)

1. Design the individual classifiers $D_1, \ldots, D_L$ using the labeled data set $\mathbf{Z}$. Pick the number of regions K.
2. Disregarding the class labels, cluster $\mathbf{Z}$ into K clusters, $C_1, \ldots, C_K$, using, e.g., the K-means clustering procedure. Find the cluster centroids $\mathbf{v}_1, \ldots, \mathbf{v}_K$ as the means of the points in the respective clusters.
3. For each cluster C_j, (defining region R_j), estimate the classification accuracy of $D_1, \ldots, D_L$ using only these elements of $\mathbf{Z}$ which are in C_j. Nominate the classifier with the highest accuracy as $D_{i(j)}$.
4. Return $\mathbf{v}_1, \ldots, \mathbf{v}_K$ and $D_{i(1)}, \ldots, D_{i(K)}$.

Figure 2: Training of the Clustering and selection method

Clustering and selection is guaranteed by design to give at least the same training accuracy as the best individual classifier D^*. We can use a statistical test (e.g., χ^2 or paired t-test) to find out whether the first and the second best classifiers are significantly different in R_j. If yes, then we can pick the winner to label all $\mathbf{x}$'s in R_j. If not, we should use another model, e.g., average or majority vote, so that classifiers that are significantly better than the others have a vote in labeling $\mathbf{x}$. Verikas et al. [8] propose various heuristic schemes for combining classifiers with region-specific parameters.

Clustering and selection (operation)

1. Given the input $\mathbf{x} \in \Re^n$, find the nearest cluster center from $\mathbf{v}_1, \ldots, \mathbf{v}_K$, say, $\mathbf{v}_j$.
2. Use $D_{i(j)}$ to label $\mathbf{x}$, i.e., $\mu_k(\mathbf{x}) = d_{i(j),k}(\mathbf{x})$, $k = 1, \ldots, c$.

Figure 3: Operation of the Clustering and selection method

4 Experimental illustration

Experiments with three data sets have been carried out:

1. Cone-torus data. This is a three class dataset with 400 2-d points generated from three differently shaped distributions: a cone, half a torus, and a normal distribution with prior probabilities 0.25, 0.25, and 0.5, respectively. This data set is available on http://www.bangor.ac.uk/~mas00a/Z.txt. A separate data set for testing with 400 more points generated from the same distribution is also available as the file Zte.txt.

2. Normal-mixtures data. This dataset is used in [7] for illustrating classification techniques. The training data consists of 2 classes with 125 2-d points in each class. The points in each class come from a mixture of two normal distributions with the same covariance matrices. The data set is available on http://www.stats.ox.ac.uk/~ripley/PRNN/. A testing set containing 1000 more points drawn from the same distribution is also provided.

3. Phoneme data. This set is from the ELENA database. It consists of 5404 five-dimensional vectors characterizing two classes of phonemes: nasals (70.65 %) and orals (29.35 %).

In all experiments the training and testing parts are formed as follows. With the Cone-torus and Normal-mixtures data, the two parts are used for training and for testing, as designated. With the Phoneme data, the *first 500* elements are used for training, and the remaining 4904 elements are used for testing.

We trained five multi-layer perceptron (MLP) neural networks as the classifiers $D_1, \ldots, D_5$. Each MLP had one hidden layer with 20 nodes in it. The hidden and the output nodes had sigmoidal activation functions, so the outputs $d_{i,j}(\mathbf{x})$ were in the interval $[0, 1]$. Fast backpropagation was used to train the neural networks (Matlab Neural Network Toolbox). To examine the effect of under- and overtraining, three protocols were used terminating the training after 100, 300 and 2000 iterations (epochs), respectively. The fusion schemes studied here were: (1) MAJ. Majority vote, (2) NB. Naive Bayes [10]; (3) BKS. Behavior Knowledge Space method [3]; (4) MAX. Maximum rule; (5) MIN. Minimum rule; (6) AVR. Average rule; (7) PRO. Product rule; and (8) **CS**. Clustering and selection. For reference we also used: (9) SB. The single best (individual) accuracy and (10) OR. The oracle. The *oracle* works as follows: assign the *correct* class label to $\mathbf{x}$ iff at least one individual classifier produces the correct class label of $\mathbf{x}$.

The results with the three data sets are displayed in Table 1. In all cluster-and-selection experiments we used c-means procedure with $K = 8$ (clusters). Clustering and selection was performed 10 times because the c-means procedure depends on the initialization. The best training result from the 10 runs was identified and the corresponding testing result is shown in the table. When the lowest training error tied, the corresponding testing error was taken as the averaged testing errors of the tied runs.

It is difficult to judge the performance of the combination schemes by the percentages in Table 1. A better characteristic can be calculated using the order of the error rates. Thus, combination schemes with a consistently good behavior can be detected. Table 2 gives the rank scores corresponding to the results in Table 1. The best combination scheme (lowest error) has the highest rank (11 for each experiment in our case) and the worst scheme will have the lowest rank. Not surprisingly, the best score is achieved by the hypothetical oracle.

5 Conclusions

The first conclusion from the tables is that the CS combination model has the best rank score, 54.5 (excluding the oracle which is only an abstraction). Next best robust competitor has been the majority vote with total rank of 51. The simple fusion methods which are often highly accurate did not fare well in these experiments. The favorite amongst them here has been the average. The BKS method is notorious for its overtraining and this is the reason that it only just outperformed the single best individual NN. Interestingly, BKS did not give the smallest training error of all combination methods either (not shown here). The winner was again the Clustering and selection model. The Naive Bayes method relies on some independence assumptions which might or might not hold and this determines the result to a high degree.

Table 1 shows that under- and overtraining depend on the data. With the Cone-torus data with $c = 3$ classes, 100 epochs have not been sufficient to train all 5 NN's. This is the reason for the poor performance of the maximum, minimum and product rules. Although the pool $\mathcal{D}$ was highly unbalanced (containing inaccurate classifiers), the other combination schemes managed to ignore the poor NN's. Even 2000 training epochs were not sufficient to train all 5 NN's and this affected the mini-

Table 1: Testing error rates in % for 100, 300 and 2000 epochs

Method	Cone-torus			Normal-mixtures			Phoneme		
	100	300	2000	100	300	2000	100	300	2000
MAJ	21.00	12.50	11.00	9.50	10.80	11.90	25.30	17.17	17.23
NB	15.75	12.75	12.25	9.50	10.80	11.90	21.27	17.17	17.23
BKS	15.75	13.25	11.75	10.40	11.20	12.00	19.45	18.03	16.56
MAX	53.50	12.25	11.00	10.00	10.70	11.50	20.04	17.72	29.69
MIN	53.50	39.75	38.50	9.20	10.40	11.50	29.69	29.69	29.69
AVR	17.50	12.25	11.25	9.80	10.50	11.80	20.64	17.64	17.84
PR	42.00	40.25	38.75	9.60	10.40	11.80	29.69	28.28	29.69
CS	15.25	12.25	12.58	10.00	11.11	11.66	19.52	17.50	16.91
SB	15.75	13.50	11.25	9.90	11.40	11.20	19.90	17.57	17.33
OR	12.75	3.25	03.25	7.50	7.90	8.60	7.40	7.52	6.89

Table 2: Ranks of the combination schemes corresponding to the results in Table 1

Method	Cone-torus			Normal-mixtures			Phoneme			Total
	100	300	2000	100	300	2000	100	300	2000	rank
MAJ	4	6	8.5	7.5	4.5	2.5	3	8.5	6.5	51.0
NB	7	5	4	7.5	4.5	2.5	4	8.5	6.5	49.5
BKS	7	4	5	1	2	1	9	3	9	41.0
MAX	1.5	8	8.5	2.5	6	7.5	6	4	2	46.0
MIN	1.5	2	2	9	8.5	7.5	1.5	1	2	35.0
AVR	5	8	6.5	5	7	4.5	5	5	4	50.0
PR	3	1	1	6	8.5	4.5	1.5	2	2	29.5
CS	9	8	3	2.5	3	6	8	7	8	54.5
SB	7	3	6.5	4	1	9	7	6	5	48.5
OR	10	10	10	10	10	10	10	10	10	90.0

mum and the product rules. If the individual classifiers are overtrained, so is the combination. This tendency is clearly expressed with the Normal-mixtures data in Table 1. With the Phoneme data overtraining happened between the 300 and the 2000 epoch. CS cannot prevent the overtraining effect. However, as the results with the 100 epochs for the Cone-torus and Phoneme data suggest, CS is good for undertrained classifiers. The comments offered here are only based on a limited set of experiments and are not claimed to be universally valid.

References

[1] E. Alpaydin and M. I. Jordan. Local linear perceptrons for classification. *IEEE Transactions on Neural Networks*, 7(3):788–792, 1996.

[2] B.V. Dasarathy and B.V. Sheela. A composite classifier system design: concepts and methodology. *Proceedings of IEEE*, 67:708–713, 1978.

[3] Y.S. Huang and C.Y. Suen. A method of combining multiple experts for the recognition of unconstrained handwritten numerals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:90–93, 1995.

[4] R.A. Jacobs, M.I. Jordan, S.J. Nowlan, and G.E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3:79–87, 1991.

[5] L.I. Kuncheva. Change-glasses approach in pattern recognition. *Pattern Recognition Letters*, 14:619–623, 1993.

[6] L.A. Rastrigin and R.H. Erenstein. *Method of Collective Recognition*. Energoizdat, Moscow, 1981. (In Russian).

[7] B.D. Ripley. *Pattern Recognition and Neural Networks*. University Press, Cambridge, 1996.

[8] A. Verikas, A. Lipnickas, K. Malmqvist, M. Bacauskiene, and A. Gelzinis. Soft combination of neural classifiers: A comparative study. *Pattern Recognition Letters*, 20:429–444, 1999.

[9] K. Woods, W.P. Kegelmeyer, and K. Bowyer. Combination of multiple classifiers using local accuracy estimates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:405–410, 1997.

[10] L. Xu, A. Krzyzak, and C.Y. Suen. Methods of combining multiple classifiers and their application to handwriting recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, 22:418–435, 1992.