ELSEVIER

Pattern Recognition Letters 23 (2002) 183–190

Pattern Recognition Letters

www.elsevier.com/locate/patrec

Genetic feature selection combined with composite fuzzy nearest neighbor classifiers for hyperspectral satellite imagery

Shixin Yu *, Steve De Backer, Paul Scheunders

VisionLab, Department of Physics, University of Antwerp, Groenenborgerlaan 171, Antwerp, B-2020, Belgium

Received 2 October 2000; received in revised form 19 June 2001

Abstract

For high-dimensional data, the appropriate selection of features has a significant effect on the cost and accuracy of an automated classifier. In this paper, a feature selection technique using genetic algorithms is applied. For classification, crisp and fuzzy k-nearest neighbor (kNN) classifiers are compared. Composite fuzzy classifier architectures are investigated. Experiments are conducted on airborne visible/infrared imaging spectrometer (AVIRIS) data, and the results are evaluated in the paper.

Keywords: Classification; Feature selection; Hybrid systems

1. Introduction

Advances in sensor technology for earth observation make it possible to collect large numbers of spectral bands. For example, the NASA/JPL airborne visible/infrared imaging spectrometer (AVIRIS) generates image data in 224 bands simultaneously. For such high dimensionality, pattern recognition techniques suffer from the well-known curse-of-dimensionality phenomenon. This problem is resulting from the fact that the required number of labeled samples for supervised classification increases dramatically as a function of dimensionality (Fukanaga, 1990).

In this paper, we will discuss three different aspects of this problem. One way to overcome the problem is to reduce the dimensionality of the feature space. This can be done by extracting or selecting a subset of features from the total amount of features. Different feature extraction and selection techniques are proposed in the literature (branch-and-bound Narendra and Fukanaga, 1977, floating search Pudil et al., 1994, discriminant analysis Lee and Landgrebe, 1993).

A successful soft-computing technique for solving optimization problems is given by Genetic Algorithms (GAs) (Goldberg, 1989). GAs were applied to the problem of feature selection (Siedlecki and Sklansky, 1989; Kuncheva and Jain, 1999; Ishibuchi and Nakashima, 2000; Yang and Honavar, 1997) and were found to be very efficient to do so (Jain and Zongker, 1997).

Another problem with high dimensionalities is that the discrimination between classes becomes much more difficult. In pattern recognition, crisp classification is often replaced by fuzzy classification (Bezdek, 1981). In these techniques, the deci-

* Corresponding author. Tel.: +32-32180518; fax: +32-32180318.

E-mail address: shixin@ruca.ua.ac.be (S. Yu).

0167-8655/02/$ - see front matter
PII: S0167-8655(01)00118-0

sion to classify a data point is delayed as long as possible. Instead, membership values are assigned to the point as a function of the point's distance from its k-nearest neighbors (kNN) and those neighbors' memberships in the possible classes. These techniques have proven to outperform the crisp classification techniques, especially when clusters tend to overlap. In this paper, crisp and fuzzy kNN algorithms (Keller et al., 1985) are compared for the classification of hyperspectral images.

Another approach to improve classification is to combine classifiers. Achieving higher classification accuracy by combining multiple classifiers is an effective technique, and an important research topic in the literature (Xu et al., 1992; Rogova, 1994; Lam and Suen, 1995; Woods et al., 1997; Kittler et al., 1998). Although there is a rather large research body on multiple combining methods for classifiers, very little effort has been done with combining the specific classifier: the nearest neighbor classifier. In this paper, composite classifier architectures for nearest neighbor classifiers are investigated (Skalak, 1997; Wolpert, 1989).

This paper is organised as follows: in the following section, the genetic feature selection procedure is elaborated. In Section 3, the crisp and fuzzy kNN algorithms are explained. In Section 4, the composite fuzzy nearest neighbor classifiers are presented and recapitulated. In Section 5, experiments are conducted on AVIRIS data. The experimental setup is explained and results are discussed. Finally, we conclude this paper.

2. Genetic feature selection

We will follow the procedure of Siedlecki and Sklansky (1989). They designed a genetic feature selection algorithm that was found to be very efficient for high (>20) dimensionalities Jain and Zongker (1997). Unlike classical optimization procedures it does not optimize a single solution, but, instead, it modifies a population of solutions at the same time. This guarantees at least a suboptimal optimization.

A solution is represented by a finite sequence of 0's and 1's, called a chromosome. The chromosomes are allowed to 'crossover', i.e. two chromosomes exchange their parts at a randomly chosen point to create two new chromosomes. Chromosomes are also allowed to 'mutate', i.e. a small change (e.g. flipping of a bit) can be made to a chromosome. The optimization process is carried out in 'generations', where each time a population of new chromosomes is generated. Since the population size is finite, only the 'best' chromosomes are allowed to survive. A 'fitness' function is defined that allows to calculate a fitness score for each of the chromosomes.

For the problem of feature selection, a chromosome has length d, the total number of features. A '1' stands for a selected feature, whereas a '0' stands for a rejected feature. There are two ways of optimizing a string. One way is to minimize the classification error rate. This, however, will not necessarily minimize the number of selected features. Better is to define a threshold error rate t, and to find the string with the lowest number of selected features that leads to an error rate e, lower than t. A fitness function is defined that takes this into account:

$$f(a_i) = J(a_j) - (\langle J(a_i)\rangle - c\Delta J(a_i)),$$

where $\langle\cdot\rangle$ and Δ are the mean and standard deviation over the population, c is a small positive constant which assures that $\min f(a_i) > 0$, i.e. even the least fit chromosome is given a chance to reproduce. The score $J(a)$ of a string a is given by

$$J(a) = l(a) + p(e(a)),$$

where $l(a)$ is the 'length' (=number of '1') of string a, and $p(e)$ is a penalty function for the obtained error rate e. If e is below the threshold error rate t, $p(e)$ is negative, and if e grows larger than t, $p(e)$ grows rapidly:

$$p(e) = \frac{\exp(e-t)/m - 1}{\exp(1) - 1},$$

with m as a small scaling parameter (about 1%).

3. Fuzzy kNN classification

In the feature selection procedure, the fitness function to be calculated includes the calculation of the classification error rate. Therefore, a classifier is to be designed. In this paper, we want to compare 'crisp' and 'fuzzy' classifiers.

The nearest neighbor rule is one of the oldest and simplest methods for performing non-parametric classification. Despite its simplicity, it has many advantages, e.g. it may give competitive performance compared to many other methods. An easy and effective way to calculate the classification error rate is by the "leave-one-out" procedure. Hereby, each time the complete training set, but one, is used, and the left out training sample is used for testing. By doing this for each training sample separately, the classification error rate can be evaluated.

A fuzzy kNN classifier was designed by Keller et al. (1985). Hereby, class memberships are assigned to the sample, as a function of the sample's distance from its kNN training samples:

$$u_i(x) = \frac{\sum_{j=1}^{k} u_{ij}\left(1/\|x - x_j\|^{2/(w-1)}\right)}{\sum_{j=1}^{k}\left(1/\|x - x_j\|^{2/(w-1)}\right)},$$

with w as a scaling parameter between 1 and 2. The memberships of the training samples u_{ij} can be defined in several ways. The 'crispest' way is to give them complete membership in their own class and nonmembership in all other classes. A more 'fuzzy' alternative is to assign the training samples' memberships based on the distance from their class mean. After calculating the memberships for the test sample, it is assigned to the class with highest membership. In our experiments, we have found that the second approach leads to the best results.

4. Composite fuzzy nearest neighbor classifiers

We follow Skalak (1997) discussion on composite nearest neighbor classifiers. Of the many architectures for classifier combinations till now, there are three primary architectures for combining classification algorithms: (1) *Stacked Generalization* (Fig. 1); (2) *Boosting* (Fig. 2); (3) *Recursive Partitioning* (Fig. 3).

Wolpert (1989) was probably the first to discuss the idea of Stacked Generalization in its full generality. Stacked Generalization assumes that a set of n level-0 (component) learning algorithms, a level-1 learning (combining) algorithm, and a training set of classified instances have been given. It is a recursive layered structure for combining classifiers, where at each layer the classifiers are used to combine the output of the classifiers just under that layer. Boosting is due to Schapire (1990). The goal of boosting is to increase the accuracy of a given algorithm on a given distribution of training in-

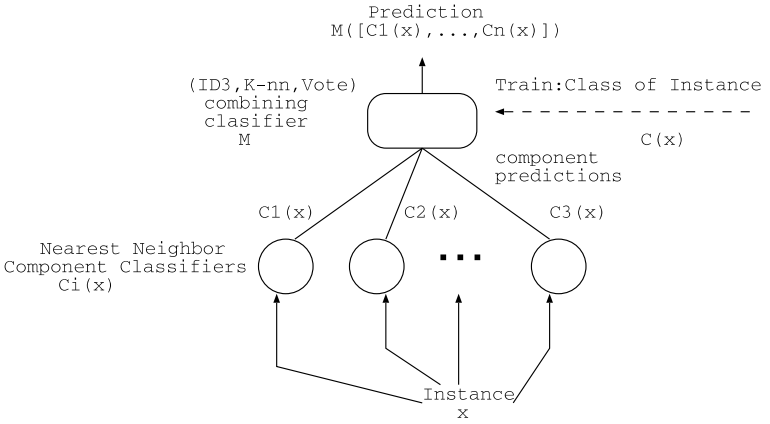

Fig. 1. Stacked nearest neighbor classifier architecture.

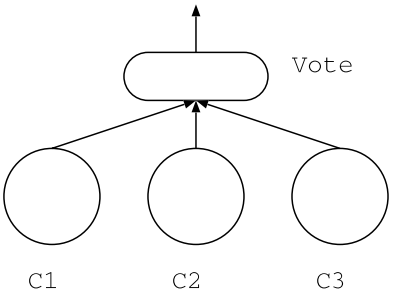

Fig. 2. Boosting architecture. Classifier C_1 is taken as given, while C_2 and C_3 are additional component classifiers.

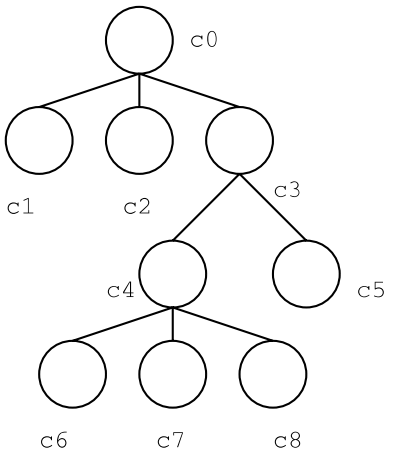

Fig. 3. An example of a recursive partitioning architecture. Each of the component classifiers, from c_0 to c_8, applies only to a particular region of the instance space.

stances. It successively creates complementary component classifiers by filtering the training set. Recursive Partitioning algorithms use a divide-and-conquer strategy to partition a space into regions that contain instances of only one class.

Skalak (1997) investigated one specific boosting architecture as shown in Fig. 4. It is a two-layer architecture, consisting of level-0 and level-1 classifiers. The level-0 classifiers consist of two classifiers: a base classifier (say C_0), i.e. a full nearest neighbor classifier, which uses all instances as prototypes, and a complementary classifier (say C_1), which is a minimal nearest neighbor classifier (Skalak, 1994), storing only one prototype per class. In this work, the complementary nearest neighbor classifier C_1 is obtained through the following procedure:

1. randomly sample n sets of s instances (with replacement) from the training set T, where s is the number of classes exposed in T, one instance is drawn from each class;
2. use each set as a prototype set to construct a nearest neighbor classifier;
3. classify all instances in T using each of these n classifiers;
4. choose the classifier with highest classification accuracy on T as the complementary classifier C_1.

For the level-1 combining algorithm, the decision tree algorithm ID3 (Quinlan, 1986) is used. For each original training instance $x \in T$ with class S_i, a level-1 training instance is created: $(C_0(x), C_1(x), S_i)$. So, for example, let x be a level-0 instance, if C_0 predicts class A when applied to instance x, while C_1 predicts class B when applied to instance x, then the level-1 feature representation for x becomes (A, B). The *entire* level-1 representation for x also includes the class of x (say

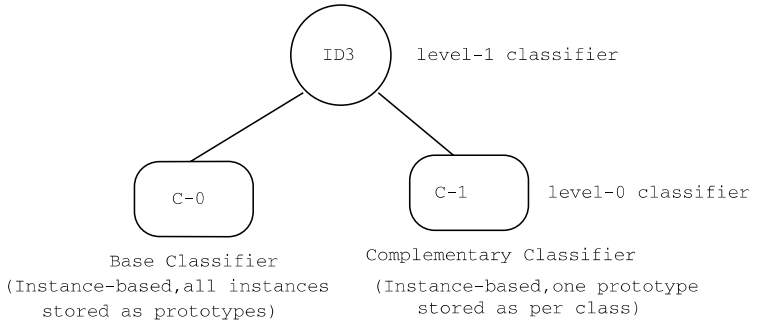

Fig. 4. A composite architecture.

A), i.e. the level-1 representation is actually (A, B, A). The set of these three-tuples of class samples is the training set used to train the level-1 learning algorithm ID3.

ID3 is a greedy algorithm that grows the tree top-down, at each node selecting the attribute that best classifies the local training examples. This procedure continues until the tree perfectly classifies the training examples, or until all attributes have been used. Although ID3 is no longer considered as a state-of-the-art decision tree, we use it as the level-1 combining algorithm. All the level-1 features are symbolic, the implementation of ID3 uses the same feature selection metric as described by decision tree C4.5 (Quinlan, 1993), a descendant of ID3.

We will apply the above mentioned architecture for our AVIRIS dataset. In the level-0 layer, the nearest neighbor classifiers are replaced by their fuzzy counter-parts, as the fuzzy approach outperforms the crisp approach discussed in the earlier section.

5. Experiments and discussion

Experiments were conducted on an AVIRIS dataset, containing 220 bands of 145×145 pixels, that is downloadable from http://dynamo.ecn.purdue.edu/~biehl/MultiSpec/documentation.html, along with a groundtruth image, containing 16 classes.

In the first experiment, the genetic feature selection technique is evaluated. For this, the minimal number of obtained features is plotted in function of the number of generations. The parameters t and m are set so that the classification error is about 10%. In Fig. 5, experimental results are displayed for a 3-class problem (classes 2, 3 and 8), with 100 data points for each class. Several experiments starting with different numbers of bands are conducted. On the plot, the numbers of bands are 50, 100, 150 and 220, respectively, i.e. the first 50 bands, 100 bands, 150 bands of the dataset and the full 220-band data.The population size was 100. The crossover rate usually assumes high values, close or equal to one, while the mutation rate is typically small (Siedlecki and Sklansky, 1989). The crossover rate is high to allow to produce an offspring that is more optimal than its parents. A 100% crossover rate would, however, disrupt any good solution. In our experiment, crossover and mutation rates were set to 90% and 1%, respectively. For classification, the fuzzy 5NN algorithm was applied. From the plot, one can

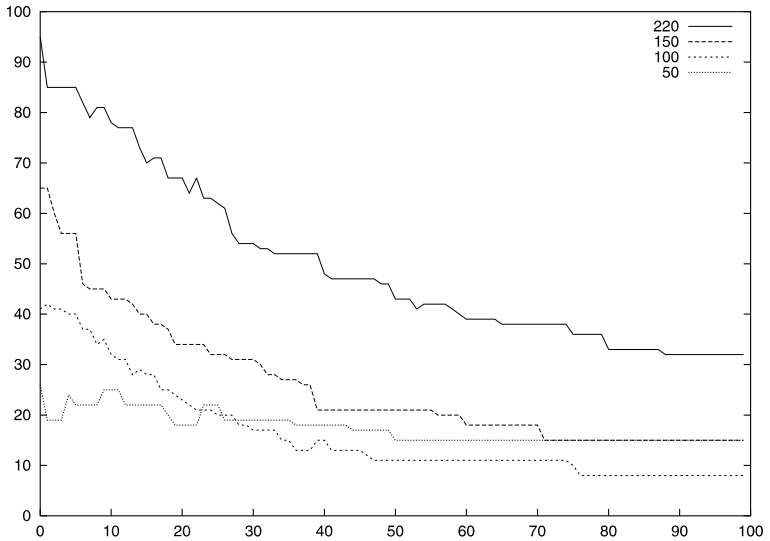

Fig. 5. Number of obtained features versus number of generations for the genetic feature selection on 50, 100, 150 and 220 bands, respectively.

observe that the number of obtained features decreases with the number of generations. The classification errors were about constant over all the curves (about 10%). In the beginning the reduction is very fast, but after about 50 generations convergence becomes slow.

In experiment 2, crisp and fuzzy kNN classification are compared for high dimensionality. For this, the classification performance is evaluated as a function of the number of applied features that is gradually augmented by performing the floating forward feature selection technique (Pudil et al., 1994). The results reported in Fig. 6 refer to a 5-class (classes 2, 3, 8, 10 and 11) classification task, with 100 data point for each class, and repeated 10 times using different data points (the obtained error bars are plotted). One can observe a systematic improvement of about 5% using the fuzzy approach. For dimensions >30, the classification performance goes down, because of the curse of dimensionality phenomenon (Landgrebe, 2000), and results become unreliable.

Similar results can be obtained by using the genetic feature selection technique. Since the fitness function optimizes the classification error and the number of features simultaneously, no continuous curve can be obtained. To be able to plot the classification result in function of the number of features, the parameters t and m are varied. On the figure, we have plotted about 20 obtained

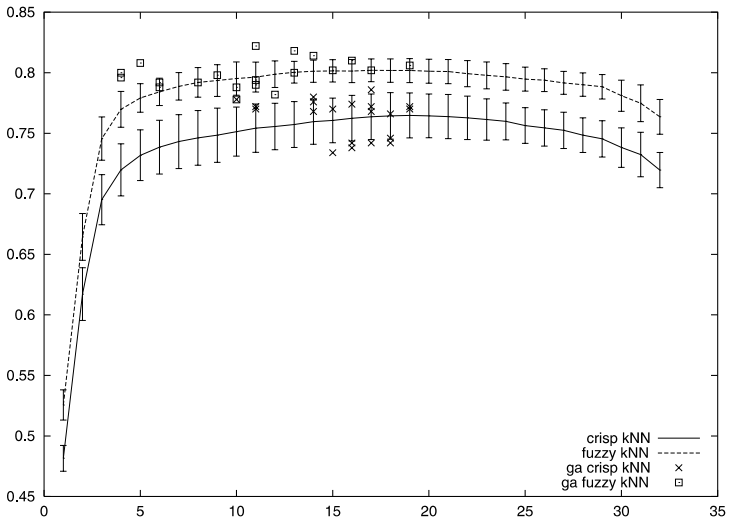

Fig. 6. Classification performance versus dimensionality for a 5-class problem using crisp and fuzzy kNN. The curves are obtained by using floating forward feature selection, the individual points by using genetic feature selection.

band numbers	NB	5NN	Fuzzy 5NN	C4.5	composite
50	29.51	15.31	13.66	14.88	11.22
100	29.51	13.85	9.27	10.49	7.07
150	29.76	13.85	6.34	11.74	5.85
200	28.54	13.80	6.10	12.44	5.85

Fig. 7. Comparison of error rate among Naive–Bayes (NB), a single kNN ($k = 5$), a single fuzzy kNN ($k = 5$), decision tree C4.5 and composite fuzzy nearest neighbor classifiers on different bands of the data set.

results using genetic feature selection with the crisp kNN and about 20 results using genetic feature selection with the fuzzy kNN algorithm. One can observe that the points follow the curves, obtained by using floating forward feature selection, some results are even better (up to 2%).

We can conclude that the genetic feature selection technique is a useful alternative for the floating search feature selection technique. The use of the fuzzy kNN algorithm improves results over crisp kNN. This holds for the floating search results as well as for the genetic selection results.

The third experiment was carried out on the composite classifiers architecture as explained in Section 4. In Fig. 7 experimental results are displayed for a 3-class problem (classes 2, 3, and 8), with 100 instances per class. The experiments were performed on dataset with different numbers of bands. The numbers of bands are 50, 100, 150 and 200, respectively, i.e. the first 50 bands, 100 bands, 150 bands and 200 bands of the full dataset. In our experiment, we used $n = 100$ samples, $k = 1$ (fuzzy 1NN) for C_1, $k = 5$ (fuzzy 5NN) for C_0. We compared the performance of the composite structure shown in Fig. 4 with that of C4.5, Naive–Bayes (NB) (Cestnik, 1990), a single kNN ($k = 5$) and a single fuzzy kNN ($k = 5$). The justification for comparing with C4.5 is based on the fact that the level-1 combination algorithm in Fig. 4 is ID3, and the C4.5 is a further extension of ID3.

For evaluation, 10-fold cross-validation was used. From the comparison shown in Fig. 7, we can see that the composite architecture using fuzzy nearest neighbor classifiers outperforms both the C4.5 decision tree and the single 5NN classifier (an average of about 6% improvement) in terms of error rate. The difference with the single fuzzy 5NN classifier is small, and diminishes for high dimensionality. Although some researchers (Kohavi and John, 1997) noted that NB's accuracy is superior over C4.5 in some real dataset, this conclusion did not show in our dataset.

6. Summary and conclusions

In this paper, the genetic feature selection technique is applied to high dimensional remote sensing data, the effectiveness of its use is presented. In contrast to the extensive research on combining classifiers in the literature, there is few work on the combination of nearest neighbor classifiers. In this paper, composite fuzzy nearest neighbor classifiers are investigated and their superior performance is well justified by our experimental results.

References

Bezdek, J., 1981. Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York.

Cestnik, B., 1990. Estimating probabilities: a crucial task in machine learning. In: Proc. of the European Conf. on Artifical Intell., Stockholm, Sweden, pp. 147–149.

Fukanaga, K., 1990. Introduction to Statistical Pattern Recognition. Academic Press, San Diego, CA.

Goldberg, D., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading, MA.

Ishibuchi, H., Nakashima, T., 2000. Multi-objective pattern and feature selection by a genetic algorithm. In: Proc. Conf. on Genetic Evolutionary Computat., Las Vegas, Nevada, USA, July 8–12, pp. 1069–1076.

Jain, A., Zongker, D., 1997. Feature selection: evaluation, application, and small sample performance. IEEE Trans. Pattern Anal. Machine Intell. 19 (2), 153–158.

Keller, J.M., Gray, R., Givens, J.A.J.R., 1985. A fuzzy k-nearest neighbor algorithm. IEEE Trans. Systems Man Cybernet. 15 (4), 580–585.

Kittler, J., Hatef, M., Duin, R.P.W., Matas, J., 1998. On combining classifiers. IEEE Trans. Pattern Anal Machine Intell. 20 (3), 226–239.

Kohavi, R., John, G., 1997. Wrappers for feature subset selection. Artificial Intelligence 97 (1–2), 273–324.

Kuncheva, L.I., Jain, L.C., 1999. Nearest neighbor classifier: simultaneous editing and feature selection. Pattern Recognition Lett. 20, 1149–1156.

Lam, L., Suen, C.Y., 1995. Optimal combination of pattern classifiers. Pattern Recognition Lett. 16, 945–954.

Landgrebe, D., 2000. Information extraction principles and methods for multispectral and hyperspectral data. In: Chen, C.H. (Ed.), Information Process. Remote Sensing. World Scientific, Singapore.

Lee, C., Landgrebe, D., 1993. Feature extraction based on decision boundaries. IEEE Trans. Pattern Anal. Machine Intell. 15 (4), 388–400.

Narendra, P., Fukanaga, K., 1977. A branch and bound algorithm for feature subset selection. IEEE Trans. Comput. C-26 (9), 917–922.

Pudil, P., Novovicova, J., Kittler, J., 1994. Floating search methods in feature selection. Pattern Recognition Lett. 15, 1119–1125.

Quinlan, J.R., 1986. Induction of decision trees. Machine Learning 1, 81–106.

Quinlan, J.R., 1993. C4.5: Programs for Machine learning. Morgan Kaufmann, San Marteo, CA.

Rogova, G., 1994. Combining the results of several neural network classifiers. Neural Networks 7, 777–781.

Schapire, R.E., 1990. The strength of weak learnability. Machine Learning 5, 197–227.

Siedlecki, W., Sklansky, J., 1989. A note on genetic algorithms for large-scale feature selection. Pattern Recognition Lett. 10, 335–347.

Skalak, D.B., 1994. Prototype and feature selection by sampling and random mutation hill climbing algorithms. In: Proc. 11th Internat. Conf. on Machine Learning, New Brunswick, NJ.

Skalak, D.B., 1997. Ph.D Thesis: Prototype selection for composite nearest neighbor classifiers. University of Massachusetts, Amherst.

Wolpert, D.H., Stacked Generalization, LA-UR-90-3460 Complex Systems Group, Theoretical Division, and Center for Non-linear Studies. MS B213, LANL, Los Alamos, NM.

Woods, K., Kegelmeyer, W.P., Bowyer, K., 1997. Combination of multiple classifiers using local accuracy estimates. IEEE Trans. Pattern Anal. Machine Intell. 19, 405–410.

Xu, L., Krzyzak, A., Suen, C.Y., 1992. Methods of combining multiple classifiers and their application to handwriting recognition. IEEE Trans. Systems Man Cybernet. 22, 418–435.

Yang, J., Honavar, V., 1997. Feature subset selection using a genetic algorithm. In: Proc. Second Conf. on Genetic Programming.