A Comparison of Rule-Based, K-Nearest Neighbor, and Neural Net Classifiers for Automated Industrial Inspection

Tai-Hoon Cho*, Richard W. Conners*, and Philip A. Araman**

* The Spatial Data Analysis Laboratory
Virginia Polytechnic Institute & State University
Blacksburg, VA 24061

** Brooks Forest Products Center
Virginia Polytechnic Institute & State University
Blacksburg, VA 24061

Abstract

Over the last few years the authors have been involved in research aimed at developing a machine vision system for locating and identifying surface defects on materials. The particular problem being studied involves locating surface defects on hardwood lumber in a species independent manner. Obviously, the accurate location and identification of defects is of paramount importance in this system. In the machine vision system that has been developed, initial hypotheses generated by bottom-up processing for defect labeling are verified using top-down processing. Thus, the label verification greatly affects the accuracy of the system. For this label verification, a rule-based approach, and a k-nearest neighbor approach, and a neural network approach have been implemented. An experimental comparison of these approaches together with other considerations have made the neural net approach the preferred choice for doing the label verification in this vision system.

1. Introduction

Over the last few years the authors have been involved in research aimed at developing a machine vision system for locating and identifying surface defects on materials. The particular problem being studied involves locating surface defects on hardwood lumber in a species independent manner. Obviously, the accurate location and identification of defects is of paramount importance in this system. In the machine vision system that has been developed, initial labeling hypotheses generated by bottom-up processing are verified using top-down processing. Thus, the label verification operation greatly affects the accuracy of the system. For this label verification, a rule-based approach, and a *k* - nearest neighbor approach, and a neural network approach have been implemented. These approaches were tested and compared on the hardwood lumber inspection problem. In what follows the three basic approaches will be described. The relative utility of each will be presented. And finally, a description of the test performed will be discussed.

The main function of the machine vision system is to detect undesirable "defects" that can appear on the surface of the material being inspected. Conceptually, the vision system consists of two modules: the low-level segmentation module and the high-level recognition module [1]. The low-level module performs segmentation of the input image into "homogeneous" regions and extracts *basic* region properties from each of the resultant regions.

The high-level module performs the scene analysis task, i.e., recognizes defect areas. Basically, this module consists of the following components: a focus of attention mechanism, defect detection procedures, verification using contextual information or spatial constraints, and resolution of multiple labels. A focus of attention mechanism is used to screen candidate regions in an effort to determine the type or types of defect each region might represent. Each defect detection procedure is designed to detect a particular type of defect. Each defect detection procedure operates completely independent of the other defect detection procedures. Each defect detection procedure consists of an initial labeling step and a label verification step.

The initial labeling of regions is performed based on the basic region properties, region adjacency information, a priori knowledge about the defect type under consideration, and, if needed, new features extracted directly from the original image. This initial labeling step produces the sets of connected regions, called DEFECT_OBJECTs. A DEFECT_OBJECT is a set of connected regions all of which have been assigned the same defect label by a defect detection procedure. The label verification step tries to verify the assigned label of each DEFECT_OBJECT generated in the initial labeling step. This step is necessitated

by the fact that a segmented region in an image does not usually correspond to a complete defect, i.e., typically a defect is fragmented into several regions during segmentation. At this stage of the processing a basic property list is computed for each DEFECT_OBJECT. The basic properties at the level of DEFECT_OBJECT should be more representative of the actual defect than those computed from the individual regions that comprise a DEFECT_OBJECT. Besides this basic DEFECT_OBJECT property list, additional features are computed from the original image. All the features are used to verify the labeling of a DEFECT_OBJECT. Three classification methods for DEFECT_OBJECT verification have been implemented and tested. These include a rule-based approach, a *k* - nearest neighbor approach, and a neural network approach. Each approach is described in detail in the next section.

After defect detection procedures have been applied, attempts are made to further verify the labeling of each DEFECT_OBJECT by using spatial constraints among adjacent DEFECT_OBJECTs. Finally, the high-level module resolves instances where regions have been assigned multiple defect labels. A region that has been assigned multiple labels is assigned the label of a DEFECT_OBJECT whose confidence value is the highest among the confidence values of the DEFECT_OBJECTs associated with this region. Each DEFECT_OBJECT's property list is used to calculate a confidence value that the DEFECT_OBJECT has its defect label. The way this confidence value is determined depends on the approach used in the label verification step. In the rule-based approach, the confidence value of the DEFECT_OBJECT is computed using Dempster's rule of combination. In the k-nearest neighbor approach, k_i/k can be used as the confidence value, where k_i *is* the number of nearest neighbors that is the defect type of the DEFECT_OBJECT among *k* - nearest neighbors. In the neural network approach, this confidence value is determined by the output value of the neural network corresponding to the defect type of the DEFECT_OBJECT.

2. Classifiers for the Label Verification

Rule-based approaches and statistical approaches have been widely used for designing pattern classifiers. In addition, multilayer neural networks have begun to be used as classifiers in various application areas. Therefore, these three approaches have been tested in the machine vision system for industrial inspection. Among statistical approaches, a *k* - nearest neighbor classifier was selected because it does not assume any underlying distribution of data.

2.1 A Rule-Based Approach

In the rule-based approach, a series of rules or tests are applied to each DEFECT_OBECT to be verified. Each test generates confidence values that the DEFECT_OBJECT is / is not the defect type based on the test's own criterion. The confidence value of each test is determined by its own mapping function that typically uses some parameters or thresholds. This mapping function is a function of the DEFECT_OBJECT's basic properties and possibly new features extracted from the original image. Each defect detection procedure has its own series of tests. Confidence values generated by those tests are combined to yield a final confidence or belief value for the defect type using Dempster's rule of combination [2]. Only if this confidence value is greater than 0.5, is the DEFECT_OBJECT's assigned label verified.

2.2 A K-Nearest Neighbor Approach

The *k* - nearest neighbor classifier is a conventional nonparametric classifier that provides good performance for optimal values of *k*. In the *k* - nearest neighbor rule, a test sample is assigned the class most frequently represented among the *k* nearest training samples. If two or more such classes exist, then the test sample is assigned the class with minimum average distance to it. It can be shown that the *k* - nearest neighbor rule becomes the Bayes optimal decision rule as k goes to infinity [3]. However, it is only in the limit as the number of training samples goes to infinity that the nearly optimal behavior of the *k* - nearest neighbor rule is assured.

There arc at least two ways one can use the *k* - nearest neighbor classifier to perform the label verification operation. One way is to use a single *k* - nearest neighbor classifier to classify all defect types. If there are *N* defect types to be recognized, the *k* - nearest neighbor classifier must be able to handle *N+1* classes (one class for normal surface). If this is the method employed, each defect detection procedure will use this *(N+1)* -class *k* - nearest neighbor classifier to verify the labels this defect detection procedure has assigned. Hence the set of features used to do the verification is fixed and each defect detection procedure must extract these features from each DEFECT_OBJECT it creates. A defect detection procedure for recognizing defect type *i* will verify the label it has assigned to a DEFECT_OBJECT only if the *k* - nearest neighbor classifier classifies this DEFECT_OBJECT as being of defect type *i.*

Another way the *k* - nearest neighbor classifier can be used to do label verification is to use a number of two-class *k* - nearest neighbor classified. If *N* defect detection procedures are used to recognize *N* defect types, then there would be *N* two-class *k* - nearest neighbor classifiers employed. Each defect detection procedure would have its own specially tailored classifier. If a defect detection procedure is designed to recognize defect type *i*, its *k* - nearest classifier would attempt to determine whether a DEFECT_OBJECT is really of defect type *i* or not really of defect type *i*. Having such a special purpose *k* - nearest neighbor classifier inside each defect detection procedure has a number of advantages. Chief among these is that each defect detection procedure can use its own special purpose features without having to use the same feature set as all the other defect detection procedures. It is conjectured that this "pairwise" classifier should yield better results than when only one *(N+1)* -class *k* - nearest neighbor classifier is used.

2.3 A Neural Network Approach

In this approach, a multilayer feedforward artificial neural network [4] is used as the classification method instead of a *k* - nearest neighbor classifier. As in the *k* - nearest neighbor classifier there are at least two ways neural networks can be used to verify labels. One way is to use only one network to make all the verifications. As before, this approach requires all the defect detection procedures to extract the same set of features. It also means that if there are *N* defect types to be recognized the neural network must be able to handle *N+1* classes (one class for normal surface). The other approach is to use *N* two-class neural network classifiers. The advantages of this approach are the same as those described above for the set of two-class *k* - nearest neighbor classifiers. It is believed that the use of *N* two-class neural network classifiers will provide better accuracy than the use of one *(N+1)* -class neural network classifier, and as such is the recommended approach for using neural networks.

Recently, it was shown [5,6] that the multilayer perception, trained using back-propagation learning algorithm [4], approximates the optimal discriminant function defined by Bayesian theory, Specifically, the outputs of the multilayer perception approximates a posterion probability functions of the classes being trained. How closely the multilayer perception approximates a posteriori probabilities depends on the architecture of the network and the functional form of the underlying probability density function. It was also shown that a multilayer perception with more than one hidden layer has the capability of approximating any continuous mapping to any desired degree of accuracy, if a sufficient number of hidden neurons are provided [7,8,9]. These results suggest that the asymptotic behavior of the neural networks should be the same as that of the *k* - nearest neighbor method [3].

Among multilayer perceptions with hidden layers, a 3-layer perception with one hidden layer is the least complex. Further, the more hidden layers that are used, the more difficult it is to choose an appropriate number of nodes for each hidden layer. Therefore, a 3-layer feedforward artificial neural network has been chosen as a classifier for use in the label verification step employed in each defect detection procedure. Note that 2-layer neural networks without a hidden layer were excluded from the consideration because it is well known [10] that they can form only linear decision boundaries, not non-linear decision boundaries. The number of hidden neurons is determined experimentally. Enough hidden neurons must be provided so that successful learning can be obtained from the training set.

Weights in the network are obtained by using a training set. The training set consists of feature vectors computed from DEFECT_OBJECTs representing normal surface and the various defect classes. If a training sample belongs to a class *i*, then its *desired output* or *target value* is 1 for output node *i*, and 0 for all other output nodes. During training, weights are updated after one training sample is presented. This process is repeated until for each training sample, the difference between every output value and its target value is less than 0.1. (Actual target values used are 0.1 and 0.9 instead of 0 and 1, respectively, to prevent over-learning.) After training is completed, an *n* - class neural network classifier assigns class *i* to an input feature vector if output node *i* of the neural network is the highest of all the *n* output node values.

2.4. Comments on the Three Approaches

An advantage of the rule-based approach is that one can subjectively set parameters or thresholds used in each test to map the feature value obtained from the test into a confidence value, and get moderate performance even if no training data is available. Obviously, it is very desirable to incorporate information contained in training set by using it to fine-tune parameter values. If a small amount of training data is available, one can use it to tune the subjectively established values. However, if the mount of training data available is large, the ability to fine-tune parameter values can be a complex problem if one attempts to do this subjectively. This requires a significant amount of effort in knowledge engineering, if not performed automatically. Automatic setting of parameter values would be possible but not easy. Hence, the applicability of the method to industrial inspection is in doubt.

If a large training set is available, then *k* - nearest neighbor and neural network classifiers have a number of advantages over the rule-based classification method. Note that generating a large number of training samples is not difficult in most industrial inspection problems. These advantages include: 1) Both are robust and can outperform conventional parametric classifiers when the actual distribution of data is different from the assumed distribution [11]. 2) The parameters or weights of the neural network are determined automatically by a learning algorithm based on a proper training set. The *k* - nearest neighbor classifier also establishes the decision boundary automatically based on a training set. 3) Both allow incremental learning; that is, its classification performance can be incrementally refined when new training samples are added to the existing training samples.

However, there are disadvantages in using these two approaches as well. The *k* - nearest neighbor classifier has a disadvantage in that it is difficult to find an optimal value of *k* that produces the best performance for a training set with a finite number of training samples. On the other hand, the training of a neural network typically requires a large number of presentations of the training set. Another difficulty in using neural networks is selection of the proper number of hidden neurons. The number of hidden neurons must be carefully selected by experimentation. If too few hidden neurons are used, proper training is impeded. If too many are used, there is a performance degradation in generalization of the network, i.e., the network will perform poorly on unknown samples, even if it works perfectly on the training samples. Note that these disadvantages for both approaches really only represent computational problems in the training phase.

One advantage of the neural network classifier over *k* - nearest neighbor classifier is the speed of classification. The *k* - nearest neighbor classifier is computationally complex. To classify a test sample, the *k* - nearest neighbor classifier requires that the distances between the test sample and each stored training sample be computed. This computation is proportional to the number of the training samples. This problem might be overcome by using the nearest neighbor rule on an "edited" and "condensed" set of an original training set [12]. However, it is still not certain how much the training set can be reduced without performance degradation. On the other hand, the computational complexity of the neural network classifier is proportional to the number of weights. Hence, the neural network classifier has an advantage in the computational complexity and storage complexity over the *k* - nearest neighbor classifier since a large number of training samples should be available for these classifiers. Furthermore, it allows parallel hardware implementation, which can considerably speed up classification.

3. Experimental Comparison

3.1 Setup and Implementation

The performances of the vision system using the three approaches (neural network, *k* - nearest neighbor, and rule-based) in the label verification were evaluated and compared in lumber inspection problem. The software of the machine vision system was implemented in FORTRAN 77 and C on the VAX 11/785 computer system in the Spatial Data Analysis Laboratory at Virginia Tech. This vision system was tested using an extensive image data base of rough hardwood lumber. The data base was created by digitizing a number of rough hardwood lumber boards (over 160) from 4 species, i.e., cherry, red oak, yellow poplar, and maple. An 8 inch by 8 inch area of each sample was selected and scanned. This area was digitized using a 480x512x8 bits resolution black and white camera. The same lighting conditions, camera settings, and viewing angle were employed in creating the image of each sample. Each image was shading corrected to remove any nonuniformities in lighting or sensitivity across the camera's imaging array [13].

The image data base was partitioned into two sets. One set of board images was used to construct a training set that is used in the neural network and *k* - nearest classification. The training set consisting of 100 samples was carefully constructed so that it includes most variations of each defect type. Each sample is a feature vector of a DEFECT_OBJECT verified in the label verification step. Ten features are used as elements of the feature vector. Each sample is known to belong to one of five classes: clear wood (CW), splits/checks (SP), holes (HL), wane (WN), and knots (KN). There are 20 samples for each class. Samples for clear wood are collected using DEFECT_OBJECTS that are actually clear wood. The other set of the image data base was used to test the performance of the system implemented using each approach.

Table 1 shows the number of boards of each species used in the training phase and the testing phase. Most boards used in the testing were different from those used in the training. Three cherry boards and two maple boards were used in both the training and the testing to obtain more test samples for holes and wane, respectively. However, the DEFECT_OBJECTs used in the training set was excluded in the performance evaluation. This ensured the complete independence between training and testing. In this table, the number of segmentation-failures represents the number of boards

whose images caused the complete failure in segmentation process. Table 2 shows the number of defects which appear on the boards that were used in the testing. (Of course, the defects used in the training phase were not counted in this table.)

Some comments are made as to how the three approaches were actually implemented for the lumber inspection system. Basic properties or features extracted for a region or a DEFECT_OBJECT are area, average gray level, center of mass, minimum bounding rectangle (MBR), elongatedness, perimeter, compactness, touch_background, and touch_image_boundary. The MBR is useful to locate the region or DEFECT_OBJECT in the original image or the segmented image. The elongatedness feature is used to differentiate "long" defects (e.g., splits/checks) from other defects. The compactness feature is useful for finding compact defects, e.g., holes. The perimeter feature is used to calculate a region's compactness. The touch_background feature, indicating how much portion of a region's perimeter is touching material boundary, is a helpful feature in finding wane since wane almost always appears on the edge of a board. The touch_image_boundary, indicating how much portion of a region's perimeter is touching image frame boundary, is useful because a defect region can have an abnormal shape when it is on the image boundary.

In the rule-based approach, the rules used in the defect detection procedure designed to identify defect type *i* attempt to determine whether a DEFECT_OBJECT created by this detection procedure is really a defect of type *i* or not really a defect of type *i*. As such the exact rules used vary from one defect detection procedure to another. In the *k* - nearest neighbor and the neural network approaches, a single 5-class classifier was used to classify all four defect types (one class for clear wood). A single 5-class classifier was chosen over the method using five 2-class classifiers due to much simpler training. 10 input features of the classifier are extracted for each DEFECT_OBJECT to be verified. Defect detection procedure for recognizing defect type *i* will verify the label it has assigned to a DEFECT_OBJECT only if the classifier classifies this DEFECT.OBJECT as being of defect type *i*. (The neural network employed has 10 input neurons, 10 hidden neurons, and 5 output neurons. The number of hidden neurons, 10, was determined experimentally. This was the minimum number that allowed successful learning on a training set. Successful training of the neural network took about 1500 epochs on average, i.e., an epoch is one presentation of the training set.) A fast back-propagation learning scheme [1] was used in this training. This learning scheme uses a steep sigmoid function and automatically reinitializes weights of the network when the convergence of learning is "very slow".

The 10 features used in *k* - nearest neighbor and neural net classifiers include area, average gray level, elongatedness, compactness, touch_background, touch_image.boundary, a local contrast measure, gray-level variance, absolute central moment, and average gradient magnitude. Among these features, a local contrast measure, gray-level variance, absolute central moment, and average gradient magnitude are new features used to gauge local contrast and texture characteristics. The others are the *basic* features as given above.

3.2 Classification Performance

The capabilities of differentiating each defect type from clear wood are shown in Table 3, 4, and 5. The number of defects for each defect type in these tables is the number of *true* defects. If more than half of a true defect area is recognized as one or several defect types, then this defect is assigned the most dominant defect type among defect types associated. If no clear wood area on a testing board is misclassified as being a defect, then it is considered as the correct classification of the class clear wood. If at least one clear wood area is misclassified as being a defect, then it is considered as the classification of the class clear wood as the most dominant defect type among these incorrectly labeled areas.

Note that the *k* - nearest approach is better than the neural network approach in detecting defects except knots. However, the *k* - nearest neighbor classifier has a much higher false alarm rate than the neural network, i.e., the *k* - nearest neighbor classifier misclassifies clear wood as a defect more often than the neural network. In particular, many of clear wood areas were confused with knots. The parameters or thresholds used in the rule-based approach were determined based on subjective criteria, not based on the training set. This might be the reason for the low detection accuracy of holes and wane. The detection of these defect types would be improved by fine tuning the thresholds used by each test or rule in the label verification step of the corresponding procedures. However, this process would be tedious and time-consuming. On the other hand, parameters or weights in the neural network can be obtained automatically by the learning algorithm once a training set is provided.

It is quite interesting to estimate the accuracy of the above correct classification rates of the classifiers. The confidence intervals for the true error rate of a classifier can be estimated as follows. If the true but unknown error rate of the classifier is ε, and if *k* of the *n* independent, randomly selected test samples are misclassified, then *k* has the binomial distribution

$$P(k) = \binom{n}{k} \varepsilon^k (1-\varepsilon)^{n-k}.$$

Thus, the maximum likelihood estimate $\hat{\varepsilon}$ for ε is given by the fraction of the test samples misclassified:

$$\hat{\varepsilon} = \frac{k}{n}.$$

Since $E\{k\} = n\varepsilon$ and $Var\{k\} = n\varepsilon(1-\varepsilon)$,

$$E\{\hat{\varepsilon}\} = E\{k\}/n = \varepsilon$$

$$Var\{\hat{\varepsilon}\} = Var\{k\}/n^2 = \varepsilon(1-\varepsilon)/n,$$

where $E\{.\}$ and $Var\{.\}$ are expectation and variance operators, respectively. Therefore, $\hat{\varepsilon}$ is unbiased. Given $\hat{\varepsilon}$ and n, the confidence intervals of the error rate of a classifier can be found using a table [3, p.75]. Table 6 shows 95 percent confidence intervals for error rates of the three classifiers (n = 262).

4. Conclusion

As a classifier for use in automated industrial inspection, rule-based approach, and k - nearest neighbor approach, and neural network approach were discussed. These approaches were implemented and tested for label verification in the machine vision system for hardwood lumber inspection. These test results together with other considerations have led to the selection of neural networks as the preferred method for doing the label verification in this machine vision system.

References

[1] T. Cho, *A Knowledge-Based Machine Vision System for Automated Industrial Web Inspection*, Ph.D. Dissertation, Virginia Polytechnic Institute and State University, May 1991.

[2] G. Shafer, *A Mathematical Theory of Evidence.* Princeton, NJ: Princeton Univ. Press, 1976.

[3] R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis*, New York: John Wiley & Sons, 1973.

[4] D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing. Exploration of the Microstructure of Cognition, vol.1: Foundations*, D.E. Rumelhart and J.L. McClelland, Eds. Cambridge, MA: MIT Press, 1986.

[5] D.W. Ruck, S.K. Rogers, M. Kabrisky, M.E. Oxley, and B.W. Suter, "The multilayer perception as an approximation to a Bayes optimal discriminant function; *IEEE Trans. on Neural Networks*, vol.1, no.4, pp.296-298, 1990.

[6] E.W. Wan, "Neural network classification: A Bayesian interpretation," *IEEE Trans. on Neural Networks*, vol. 1, no.4, pp.303-305, 1990.

[7] G. Cybenko, "Approximation by superposition of a sigmoid function," *Mathematics of Controls, Signals, and Systems*, vol.2, pp.303-314, 1989.

[8] K. Funahashi, "On the approximate realization of continuous mappings by neural networks," *Neural Networks*, vol.2, pp.183-192, 1989.

[9] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators: *Neural Networks*, vol.2, pp.359-366, 1989.

[10] R.P. Lippman, "Art introduction to computing with neural nets," *IEEE ASSP Magazine*, vol.4, no.2, pp.4-22, 1987.

[11] W.Y. Huang and R.P. Lippman, "Comparisons between neural net and conventional classifiers," *Proc. IEEE First Int. Conf. Neural Networks*, vol.IV, Piscataway, Nj: IEEE, pp.485-493, 1987.

[12] P.A. Devijver and J. Kittler, *Pattern Recognition; A Statistical Approach*, Englewood Cliffs, NJ: Prentice Hall, 1982.

[13] A.A. Sawchuk, "Real-time correction of intensity norlinearities in imaging system," *IEEE Trans. Comp.*, vol.26, no.1, pp.34-39, 1977.

TABLE 1
The Number of Rough Boards of Each Species
Used in Training and Testing

	Cherry	Red Oak	Yellow Poplar	Maple	Total
# boards in the data base	41	52	26	48	167
# segmentation failure	2	4	1	3	10
# boards used in training	19	16	5	25	65
# boards used in testing	23	32	20	21	96

TABLE 2
The Number of Defects on Boards
of Each Species Used in Testing

	Cherry	Red Oak	Yellow Poplar	Maple	Total
Split/Check	9	11	5	2	27
Hole	28	4	0	1	33
Wane	1	8	5	3	17
Knot	24	25	22	18	89

TABLE 3
Differentiation of Each Class
Using the Neural Network Approach

		Assigned Class						
		CW	SP	HL	WN	KN	Total	% Correct
True Class	CW	86	3	1	0	6	96	89.6 %
	SP	5	21	0	0	1	27	77.7 %
	HL	8	1	21	0	3	33	63.6 %
	WN	4	0	0	13	0	17	76.5 %
	KN	14	2	0	2	71	89	79.8 %
	Total	117	27	22	15	81	262	
Overall Correct Classification (%)								80.9 %

TABLE 4
Differentiation of Each Class
Using the *k* - Nearest Neighbor Approach (*k* = 5)

		Assigned Class						
		CW	SP	HL	WN	KN	Total	% Correct
	CW	56	2	1	0	37	96	58.3 %
True	SP	4	22	1	0	0	27	81.5 %
Class	HL	4	0	27	0	2	33	81.8 %
	WN	1	0	0	16	0	17	94.1 %
	KN	15	3	4	3	64	89	71.9 %
	Total	80	27	33	19	103	262	
				Overall Correct Classification (%)				70.6 %

TABLE 5
Differentiation of Each Class
Using the Rule-Based Approach

		Assigned Class						
		CW	SP	HL	WN	KN	Total	% Correct
	CW	72	7	5	0	12	96	75.0 %
True	SP	2	20	0	0	5	27	74.1 %
Class	HL	6	0	22	0	5	33	66.7 %
	WN	4	0	0	11	2	17	64.7 %
	KN	20	0	2	0	67	89	75.2 %
	Total	104	27	29	11	91	262	
				Overall Correct Classification (%)				73.3 %

TABLE 6
95 Percent Confidence Intervals of
the Classification Error Rate

Classifier	$\hat{\varepsilon}$	Confidence Interval
Neural Net	0.191	(0.15, 0.27)
K-Nearest Neighbor	0.294	(0.24, 0.37)
Rule-Based	0.267	(0.22, 0.35)

Proceedings of the

IEEE/ACM International Conference on Developing and Managing Expert System Programs

September 30- October 2,1991 *Washington, D.C.*

Editors

Jerald Feinstein, Elias Awad,
Larry Medsker, and Efraim Turban

Sponsored by

IEEE Computer Society Technical Committee on
Applications of Expert Systems

ACM Special Interest Group on Business Information Technology

In cooperation with

The American University

George Washington University

International Association of Knowledge Engineers

1951-1991

IEEE Computer Society Press
Los Alamitos, California

Washington • Brussels • Tokyo