

## RCE CLASSIFIERS: THEORY AND PRACTICE

---

MICHAEL J. HUDAK

38 Oliver Street, Binghamton, New York 13904

Restricted Coulomb Energy (RCE) classifiers, as described by Scofield et al. (1988), are shown to have a conceptual relationship with hyperspherical classifiers developed in the 1960s by Batchelor (1974). These classifiers are also shown to share similarities with networks of localized receptive fields and with psychological models of concept formation. Next, the performance of some RCE classifiers is examined. The ability of a trained RCE classifier to generalize to new instances is compared with that of several well-known classifiers. Then, four previously unexamined aspects of RCE classifiers are investigated empirically: (1) the influence of potential wells on training rate, (2) the influence of potential wells on storage requirement, (3) the influence of potential wells on generalization to new instances, and (4) rejection of an instance from an unknown class. Modifications of a traditional RCE classifier improve average correct classification at generalization from 83.2 to 90.7% without significant change in computational cost. By comparison, a nearest-neighbor performs at 93% and a feed-forward multilayer neural network at 88.4% on the same data. Surprisingly, when the improved RCE network is compared with its underlying adaptive nearest-neighbor component, one finds that the incorporation of potential wells into the RCE classifier does not reduce training time or instance storage requirement, nor does it improve generalization to new instances.

### INTRODUCTION

Pattern classifiers constructed from the union of hyperspherical decision regions have been studied since the early 1960s. The earliest work (e.g., Cooper, 1962, 1966) was primarily theoretical in nature. Later empirical work by Batchelor (1974) compared the shape of decision regions for hyperspherical classifiers with those of other nonparametric classifiers. Not until the work of Lee (1989) and Lee and Lippmann (1990) was there a

thorough empirical examination of hyperspherical classifiers and their special case, Restricted Coulomb Energy (RCE) classifiers.

Hyperspherical classifiers have attracted attention for several reasons and have been viewed in several ways, as demonstrated by the following list.

Although resembling the nearest-neighbor classifier (Duda and Hart, 1973) in its storage of prototypical patterns, a hyperspherical classifier is typically more conservative of storage (Batchelor, 1974: 49) and can abstain from classifying a pattern from an unknown category (Li, 1988: 21).

Although a hyperspherical classifier is typically trained by repeated presentation of training instances, as is required in backpropagation training of feedforward neural networks, hyperspherical classifiers typically require considerably fewer training epochs (complete presentations of training instances) than do neural networks (Lee, 1989; Lee and Lippmann, 1990).

A hyperspherical classifier has been viewed as an "exemplar-based" model of biological memory in which memory is constructed from representation of experiences (Kanerva, 1988).

Suitability of implementing hyperspherical classifiers in hardware has been recognized in several works (Flynn et al., 1988; Potter, 1987; Reilly et al., 1987; Rimey et al., 1986).

A form of hyperspherical classifier has been conceptualized as a neural network and termed the RCE model. This provides a high-density memory system unencumbered by the storage limitations (Hopfield, 1982) of single-layer, recurrent networks (Bachmann et al., 1987; Scofield et al., 1988).

Hyperspherical receptive fields have been used as a front end to a trainable neural network classifier (Moody and Darken, 1989, Nowlan, 1990).

This study has three major purposes: (1) to show the relationship between the RCE classifier (Scofield et al., 1988) and the hyperspherical classifier as described by Batchelor (1974); (2) to point out similarities of hyperspherical classifiers to some work in neural networks and cognitive psychology; and (3), building on the experimental work of Li (1988) and Lee (1989), to explore several practical issues that arise when RCE classifiers are applied to classification problems with real data.

## HISTORICAL OVERVIEW

### Batchelor's Compound Classifier

Types of hyperspherical classifiers were introduced by Cooper (1962, 1966) and Batchelor (1968, 1969); a summary is provided in Batchelor (1974). Like the nearest-neighbor classifier, the hyperspherical classifier is based on the storage of instances (each represented as a numeric vector called a "pattern") from known classes creating points in a metric space. The distance function defined on this space provides the basis for assigning a pattern of unknown class to a known category. Unlike the nearest-neighbor classifier, each point [also called a "locate" by Batchelor (1974)] is associated with a radius that defines the point's region of influence. In other words, each locate-radius pair defines a decision region associated with the class of the locate.

Batchelor (1974) conceived of the hyperspherical compound classifier as a layered device:

The bottom layer contains subclassifiers. Each subclassifier decides whether an input instance lies inside a hypersphere.

The top layer combines the subdecisions from the bottom layer by forming their logical union.

The formulation for the 2-class case (i.e., for a classifier capable of distinguishing two classes) is provided as an example. Assume there are  $N_1$  stored patterns in class 1:  $\vec{p}_{11}, \dots, \vec{p}_{1N_1}$ . The radii of the stored patterns are represented by  $R_{11}, \dots, R_{1N_1}$ . the  $i$ th subdecision  $U_i$  is calculated as

$$U_i = \text{sign}(R_{1i} - D_{1i}) = \begin{cases} 1, & \text{if } \|\vec{p}_{1i} - \vec{x}\|_2 < R_{1i} \\ 0, & \text{if } \|\vec{p}_{1i} - \vec{x}\|_2 = R_{1i} \\ -1, & \text{if } \|\vec{p}_{1i} - \vec{x}\|_2 > R_{1i} \end{cases} \quad (1)$$

where  $D_{1i}$  is the Euclidean distance defined by

$$D_{ij} = \left( \sum_k (p_{ijk} - x_k)^2 \right)^{\frac{1}{2}} \quad (2)$$

and  $\bar{x}$ , the instance to be classified, is represented by coordinate values  $x_1, \dots, x_k$ .

The complete decision of the classifier (i.e., the class to which the instance  $\bar{x}$  is supposed to belong) is then calculated as

$$C = \begin{cases} 1, & \text{if } \bigcup_{i=1}^{N_i} (U_i \equiv 1) \\ 2, & \text{otherwise} \end{cases} \quad (3)$$

Batchelor (1974) developed procedures that construct a hyperspherical compound classifier from a set of instances of known class. These procedures require the iterative presentation of instances until either they are all correctly classified, an equilibrium is established (i.e., further changes of radii and locates are insignificant), or a limit on iterations is exceeded. The elements of his approach are given below for the 2-class case. In following Batchelor's notation,  $T(\text{teacher})$  is the actual class to which the training instance  $\bar{x}$  belongs;  $C(\text{classifier})$  is the class of  $\bar{x}$  according to the previous decision of the classifier. One of three situations arise upon each presentation of a training instance:

1.  $C$  and  $T$  are equal. No changes are necessary.
2.  $C = 2, T = 1$  ( $\bar{x}$  lies outside all hyperspheres). Find the hypersphere closest to  $\bar{x}$ . Then move the locate of this hypersphere toward  $\bar{x}$  and increase the radius of the hypersphere.
3.  $C = 1, T = 2$  ( $\bar{x}$  lies inside any of the hyperspheres). The locate of each hypersphere whose class is 1 that encloses  $\bar{x}$  is moved away from  $\bar{x}$ , and its radius is reduced.

Batchelor's training algorithms based on these principles require incremental adjustment of two parameters in the classifier:

1. Locates (which begin as instances from the training set).
2. Radii (each of which is associated with a locate).

Determination of the step sizes for each of these increments is troublesome. Designating the incrementation parameter for locate moving as  $k$  and for radius adjustment as  $l$ , Batchelor (1974: 132-133) stated:

At the end of a long learning sequence, the classifier is in a state of dynamic equilibrium, with a small tremor on its locates and weights. The magnitude of this vibration increases with  $k$  and  $l$ . When we stop learning, we fix the classifier at some arbitrary point in this vibrating equilibrium state. The accuracy of this fixed classifier is likely to be reduced if  $k$  and  $l$  are large. On the other hand, the learning will take an unduly long time if very small values of  $k$  and  $l$  are used. . . . Certainly, we should never use larger values, although if computer time is available, smaller values can be used. It should be possible to reduce  $k$  and  $l$  progressively, in such a way that we obtain rapid learning, with a precise classifier available afterwards. We do not yet know how this should be done to achieve optimal results.

Batchelor (1974: 130–134) extended this 2-class definition to an arbitrary number of classes with an  $N$ -class classifier consisting of  $N$  2-class compound classifiers. The  $i$ th subclassifier is individually trained to separate the  $i$ th class from all others. During testing the classifiers are operated in parallel. Ambiguity in the classification is detected by recognizing the simultaneous outputs from two or more subclassifiers.

Batchelor (1974: 133) remarked that for the ranges of  $k$  and  $l$  he had tested that he “never encountered any significant changes occurring after about 5000 iterations.” (An “iteration” refers to one presentation of all training patterns, i.e., a training epoch.) Nevertheless, Batchelor (1974) suggested allowing 10,000 iterations before invoking his procedures to add more locates (growing), or removing redundant ones (pruning).

### Summary of Batchelor's Work

Details of Batchelor's procedures for training, growing, and pruning his compound classifier have been omitted here; the interested reader is referred to Batchelor (1974) for more information. The most significant characteristics of his basic classifier are summarized below.

Lack of concern for stored instances (points) as “prototypes”: Stored points can be moved in an attempt to minimize storage of instances from the training set. Hence, a stored point (called a locate by Batchelor) typically becomes unlike any instance in the training set and only defines the center of a hypersphere.

Incremental training: A locate and its radius are incrementally modified to

produce a decision region. Every locate in the classifier is a candidate for modification. Typically, up to 5000 training epochs are needed to classify all training instances correctly.

Concern for minimizing the number of locates in the classifier: New training instances are not stored unless the classification error over the training set exceeds a user-specified threshold and further improvement appears unlikely from modification of locates and radii. Furthermore, redundant or little-used locates can be removed by pruning.

No suggestion for disambiguation of classes: In the 2-class classifier formulated by Batchelor the region within hyperspheres is one class and the region outside any hypersphere is another class. Ambiguity of class membership cannot exist. For more than two classes, Batchelor proposed a method that detects ambiguity but cannot resolve it.

## RCE Networks

RCE classifiers are inspired by systems of charged particles in a three-dimensional space. I intend to show the relationship between this model, in its implementation (Scofield et al., 1988), and Batchelor's compound classifier (Batchelor, 1974). The following brief presentation leading up to the RCE model uses the notation of Bachmann et al. (1987) and Scofield et al. (1988) for the sole purpose of preventing a conflict for the reader who directly consults these works.

The classifier's name (RCE) pertains to the form of the system's Liapunov function (equation 4), which is isomorphic to the classical electrostatic potential between a positive test charge and negative charges  $Q_i$  at the sites  $\vec{x}_i$  (for three-dimensional input space and  $L = 1$ ). Viewed another way, the  $\vec{x}_i$  form a set of distinct memories in  $R^N$ .

$$\xi = -\frac{1}{L} \sum_{i=1}^m Q_i \|\vec{\mu} - \vec{x}_i\|_2^{-L} \quad (4)$$

where  $\xi$  is the electrostatic potential induced by fixed particles with charges  $-Q_i$ ,  $\vec{\mu}$  is a vector describing the network state,  $\vec{x}_i$  is the site of the  $i$ th memory for  $m$  memories in  $R^N$ , and  $L$  is a parameter related to the network size. Then  $\vec{\mu}(t = 0)$  relaxes to  $\vec{\mu}(t = T) = \vec{x}_i$  for some memory  $i$  according to

$$\dot{\vec{\mu}} = \vec{E}_{\vec{\mu}} \triangleq - \sum_{i=1}^m Q_i \|\vec{\mu} - \vec{x}_i\|_2^{-(L+2)} (\vec{\mu} - \vec{x}_i) \quad (5)$$

which "describes the motion of a positive test particle in the electrostatic field  $\vec{E}_{\vec{\mu}}$  generated by the negative fixed charges  $-Q_1, \dots, -Q_m$  at  $\vec{x}_1, \dots, \vec{x}_m$ " (Bachmann et al., 1987: 7530). Describing this system, Scofield et al. (1988: 676) stated:

The  $N$ -dimensional Coulomb energy function then defines exactly  $m$  basins of attraction to the fixed points located at the charge sites  $\vec{x}_i$ . It can be shown (Bachmann et al., 1987; Dembo and Zeitouni, 1988) that convergence to the closest distinct memory is guaranteed, *independent of the number of stored memories  $m$* , for proper choice of  $N$  and  $L$ . Equation (4) shows that each cell receives feedback from the network in the form of a scalar

$$\sum_{i=1}^m Q_i \|\vec{\mu} - \vec{x}_i\|_2^{-L} \quad (6)$$

Importantly, this quantity is the same for all cells; it is as if a single virtual cell was computing the distance in activity space between the current state and stored states. The result of the computation is then broadcast to all of the cells in the network.

Referring to an equilibrium feedforward network with similar properties described in Reilly et al. (1982), Scofield et al. (1988: 676-677) continued:

This model does not employ a relaxation procedure, and thus was not originally framed in the language of Liapunov functions. However, it is possible to define a similar system if we identify the locations of the "prototypes" of this model as the locations in state space of potentials which satisfy the following conditions

$$\xi_i = \begin{cases} -Q_i / R_o, & \text{if } \|\vec{\mu} - \vec{x}_i\|_2 < \lambda_i \\ 0, & \text{if } \|\vec{\mu} - \vec{x}_i\|_2 > \lambda_i \end{cases} \quad (7)$$

where  $R_0$  is a constant.

This form of potential is often referred to as the "square-well" potential. This potential may be viewed as a limit of the  $N$ -dimensional Coulomb potential, in which the  $1/R(L = 1)$  well is replaced with a square well (for which  $L \gg 1$ ). Equation (7) describes an energy landscape which consists of plateaus of zero potential outside of wells with flat, zero slope basins. Since the landscape has only flat regions separated by discontinuous boundaries, the state of the network is always at equilibrium, and relaxation does not occur. For this reason, this system has been called an equilibrium model. This model, also referred to as the Restricted Coulomb Energy (RCE) model, shares the property of unrestricted storage density.

The relationship between the RCE network classifier and Batchelor's compound classifier is straightforward. Both systems store training instances, and Batchelor's hypersphere radii ( $R_{ij}$ ) correspond to the RCE classifier's potential well boundaries ( $\lambda_i$ ) [Eq. (7)]. Significant differences in the formulations arise from the RCE classifier's generalization to multiple classes, the meaning of a "locate" or "memory," and the treatment of hypersphere radii and potential well boundaries. Although Batchelor requires that all stored patterns belong to one class in a given instance space, the RCE classifier provides for multiple classes in the instance space simply by storing instances from multiple classes. Furthermore, unlike Batchelor's formulation, memories in an RCE classifier cannot be modified: once an instance is stored it can be removed but not moved to another location. Finally, although Batchelor's hypersphere radii can both increase and decrease, an RCE potential well boundary can only decrease as training proceeds.

A simple training procedure for the RCE classifier is provided in Reilly et al. (1982) and is consistent with the characterization presented in Lee (1989).

### **Networks of Localized Receptive Fields (NLRF)**

Hyperspherical classifiers have a close relative in networks of localized receptive fields (NLRF) (Moody and Darken, 1989), also known as radial basis functions (Nowland, 1990). Although evidence does not exist of interaction between research on hyperspherical classifiers and NLRF, examining

their similarities and differences will provide additional perspective on the hypersphere paradigm. The work of Moody and Darken (1989) will be cited as representative of its genre.

NLRF classifiers were developed in response to extremely long training times encountered with backpropagation trained feedforward networks. In this sense the reduced training requirement of NLRF classifiers, relative to feedforward networks, parallels that of RCE networks to Batchelor's hyperspherical classifiers. Noteworthy comparisons between hyperspherical classifiers and NLRF classifiers are sixfold.

1. Terminology: "Center of receptive field" in NLRF literature is synonymous with the terms "hypersphere center," "locate," and "stored point" encountered in writings about hyperspherical classifiers, but methods for determining receptive field centers and locates are not identical.
2. Distance of influence for a "center": The influence of a hypersphere center in a hyperspherical classifier is limited by the radius of its hypersphere. In an NLRF classifier the influence of a center extends to infinity but with an intensity that diminishes exponentially with the distance from the center.
3. Selection of number of centers: Moody and Darken (1989) proposed methods for NLRF classifiers that require the a priori selection of the number of receptive field centers for the set of training instances. Typical training algorithms for hyperspherical classifiers do not have this restriction.
4. Location of centers: Moody and Darken (1989) recommended forming centers from the set of training instances with either standard *k*-means clustering or its adaptive formulation. As is typical of locate formation in Batchelor (1974), cluster centers usually do not correspond to any training instance.
5. Creation of receptive fields: Moody and Darken (1989) proposed determining the "width" of each receptive field after all the centers are chosen. Width is similar to the concept of standard deviation of a distribution. Hypersphere radius is also similar to this width, but the radius puts a hard limit on the center's influence. Furthermore, training algorithms for hyperspherical classifiers do not separate, in this manner, the processes of hypersphere center selection and radius computation.
6. Combining information from receptive fields: NLRF classifiers as described by Moody and Darken (1989) employed output weights called

receptive field amplitudes that are adjusted by a least mean square learning rule. These weights combine the responses of the receptive field functions in producing a single classification. In contrast, Batchelor's (1974) compound classifier does not contain weights. An unknown instance falling within a hypersphere is assigned the class associated with that hypersphere. If an instance falls within hyperspheres belonging to more than one class, the classifier is unable to resolve the ambiguity. RCE classifiers (Scofield et al., 1988) allow for associating a potential well "depth" with each stored point; depth is similar to the concept of weight in an NLRF.

### **Models from Cognitive Psychology**

Two models of concept representation in cognitive psychology show strong parallels to the compound classifier of Batchelor (1974) and the RCE classifier of Scofield et al. (1988). Much of this overview is taken from Matheus (1987), who outlined the models termed the "prototype view" and the "exemplar view."

The prototype view of concept formation posits that a concept is represented by a summary description in which the features need merely be characteristic of the concept instances. "The summary description is called the 'prototype' of the concept and encodes the most central features of the concept's instances. A prototype is an abstract representation and need not depict an actual instance of the concept" (Matheus, 1987: 45). Notice the similarity between prototypes and Batchelor's locates. Although a locate begins as a training instance, its ability to "drift" during training can lead to a collection of feature values unlike those possessed by any instance of the class. Note that the features in a hyperspherical classifier must be numeric whereas those of the prototype view need not be. Although the prototype model (e.g., Posner and Keele, 1970) dates from the same period as Batchelor's (1968, 1969) early work, evidence does not exist of cross-fertilization between these two fields.

Matheus (1987) pointed out major strengths and weaknesses of the prototype model. Given its relationship to Batchelor's compound classifier, these characteristics should be considered if this classifier is incorporated into a larger learning system. In favor of the prototype model, Matheus noted that it mimics human response in unclear and in typical cases. On the other hand, most prototype representations retain no information about each feature's range of values. Only information about the typical value is re-

tained. Consequently, correlational information among a concept's features is also lost.

As an alternative to the prototype model, Matheus (1987) presented what he called the exemplar view. Instead of constructing an explicit summary description for each concept, an exemplar model represents a concept as a collection of instances or exemplars (Medin and Schaffer, 1978; Medin and Smith, 1984). In a pure exemplar model (Reed, 1972) all exemplars exist as descriptions of individual instances. The relationship between the exemplar model and the RCE classifier is straightforward. Both systems store instances as collections of features that remain unchanged as learning proceeds, but, unlike the RCE classifier, these exemplar models do not surround exemplars with hyperspheres.

In contrast to the prototype model, the exemplar model provides for simpler concept formation and modification. Because a concept is represented by a collection of exemplars, modification occurs each time a new instance is added to the concept's description—no information about a feature's values is lost as new exemplars are encountered. Hence the range of encountered feature values within a concept is retained, allowing the examination of feature correlation. The major defect of the exemplar approach is its enormous storage requirement for complex concepts. Although Matheus (1987) suggested that storage can be reduced by the selective collapse of some exemplars into prototypes, he did not explain how this can be done.

### **Experimental Investigations**

The performance issues of RCE classifiers addressed in this paper rest on the experimental work of Li (1988) and Lee (1989). A brief summary of their work will provide an understanding of the questions they answered and the ones they did not explore.

Li (1988) examined the feasibility of a hypersphere-based classification system for doing low-level diagnosis of computer modules. She tested six training algorithms, elements of which are derived from Batchelor (1974). Two algorithms use incremental modification of radii; the remainder perform large-scale radius reductions, each reduction being sufficient to exclude at least one locate from a conflicting class. Half the algorithms allow locates to move, as described in Batchelor (1974); the remainder require locates to remain stationary, as in the RCE classifier of Scofield et al. (1988).

A database of 124 module signatures drawn from 35 different classes was divided into a training set of 76 patterns and a test set of 48 patterns.

Among the six algorithms, differences in the storage required are reportedly insignificant. Test performance ranged from 54 to 62% classification accuracy, defined as the predicted electronic component being either the first item on the prediction list or one of the ties for the first position. Best performance was achieved by an algorithm using incremental modification of radii and locates. Since these algorithms are sensitive to arbitrary parameters and the ordering of the training data, it is not possible to determine whether these factors, or the invariant aspects of the algorithms, account for the differences in performance. Multiple trials would have provided some insight into this matter.

Lee (1989) and Lee and Lippmann (1990) provided an evaluation of eight classifiers, including a version of Batchelor's hyperspherical compound classifier and its special case, the RCE classifier. In evaluating the hyperspherical classifier, Lee (1989: 118-121) reported the three significant points summarized below:

1. Decreasing the value by which hyperspheres were incrementally modified greatly prolonged training and improved generalization accuracy by only 1 to 2%.
2. When the number of hyperspheres was not limited by memory constraints, the improvement in generalization accuracy resulting from moving locates was insignificant. When memory was limited, moving locates significantly improved generalization accuracy, but the level achieved did not exceed that obtained with unconstrained storage.
3. Two criteria were examined for removing a locate from the classifier: the size of its hypersphere and the locate's effectiveness in correctly classifying training instances. Although a locate's effectiveness proved the better indicator of its contribution to test set classification, Lee did not report the degree of improvement.

Lee then proceeded with a closer look at the RCE classifier, which he characterized as a special case of "a generalized version of algorithms from the work of Batchelor (1974, 1978)" (Lee, 1989: 72). Lee's experiments, conducted with four databases and eight classifiers, yielded several major results about the RCE classifier. Of primary concern are the following:

The RCE classifier's generalization error exceeded the weighted average error of all classifiers by approximately 1 standard deviation on three of the databases and equaled the weighted average error in the remaining case (Lee, 1989: 133-134).

The RCE classifier trained, on the average, in 11% of the weighted average time of all the classifiers (Lee, 1989: 131).

Like the nearest-neighbor classifier, the RCE classifier stores instances from a training set. But whereas the nearest-neighbor classifier stores the entire set, the RCE classifier typically stores a minority of training set instances. Storage requirements for the RCE classifier ranged from 5.4 to 29.5% of the training instances, with the weighted average of 12.1% (Lee, 1989: 130). Values are for an "unpruned" classifier—one in which little-used instances have not been removed.

To summarize, although Lee's RCE classifier demonstrated favorable training times and memory requirements, it proved deficient in the correct classification of new instances compared with the other classifiers.

## EXPERIMENTS

### Experimental Background

Although Li (1988) and Lee (1989) performed extensive empirical studies of many hyperspherical classifiers, including RCE, several basic questions remained unanswered. Four such issues addressed in my study appear below.

1. Although Lee (1989) compared an RCE classifier with seven well-known classifiers, the experiment that would have tested the significance of hyperspheres was not performed. That experiment compares an RCE classifier to its underlying incremental nearest-neighbor component.
2. Storage requirement, training epochs, number of hyperspheres modified, and generalization to new instances are all affected by the order in which training instances are presented to a hyperspherical classifier. For Batchelor (1974), instance ordering was crucial in avoiding oscillation of locates and nonconvergence of a global error function over the training instances. Elimination of hypersphere enlargement in the RCE classifier avoids these problems but brings the issues mentioned above to prominence.
3. Lee (1989: 133–134) showed that the hyperspherical classifier is deficient in its ability to generalize, compared with the other classifiers he tested. Nevertheless, he did not propose a method for improving this performance.

4. Li (1988: 21) stated that the hyperspherical "classifier can return *no match* when the input pattern does not fall into any hypersphere in the  $n$ -space." That a hyperspherical classifier can, in theory, report that a test instance does not belong to a known category is obvious; yet this feature has not been tested for any type of hyperspherical classifier.

## Data Preprocessing

Implementation constraints for the various RCE classifiers, the nearest-neighbor, and the incremental nearest-neighbor (INN) classifier require that each feature value be mapped to one of the 256 decimal integer values (0–255). Maximum and minimum values for each feature of the training patterns were mapped to the interval end points, with intermediate values appropriately scaled. The range between end points was then used to scale the test patterns. The remaining classifiers—cascade-correlation, quickpropagation trained multilayer feedforward neural network, and decision tree classifier—do not require preprocessing of instances.

## Databases

Experiments were performed with three public domain databases: the Iris database of Fisher (1936), the Sonar database of Gorman and Sejnowski (1988), and the Congressional voting records database of 16 key votes from the 98th Congress, 2nd session, 1984. Database characteristics and details of the instance sets used for training and generalization are provided in Appendix A.

## Experimental Regime

Unless stated otherwise, the values reported for the RCE classifiers and INN are based on 100 training trials. Specifically, although trials were conducted with the same sets of training and test instances, each trial was performed with a unique random ordering of the training instances. From these trials, the average value and the standard deviation of the average are reported. The effect of training instance order on performance is thus revealed. Furthermore, expected levels of performance are established for each classifier that can then be used in meaningful comparisons.

All the RCE classifiers employ a user-specified parameter (DIST\_PCT) that reduces a hypersphere radius during training to a percentage of the

