Using Mutual Information for Selecting Features in Supervised Neural Net Learning

Article *in* IEEE Transactions on Neural Networks · August 1994

DOI: 10.1109/72.298224 · Source: IEEE Xplore

1 author:

Roberto Battiti
Università degli Studi di Trento
199 PUBLICATIONS **10,079** CITATIONS

Using Mutual Information for Selecting Features in Supervised Neural Net Learning

Roberto Battiti

Abstract—**This paper investigates the application of the *mutual information* criterion to evaluate a set of candidate features and to select an informative subset to be used as input data for a neural network classifier. Because the *mutual information* measures arbitrary dependencies between random variables, it is suitable for assessing the "information content" of features in complex classification tasks, where methods bases on linear relations (like the *correlation*) are prone to mistakes. The fact that the *mutual information* is independent of the coordinates chosen permits a robust estimation. Nonetheless, the use of the *mutual information* for tasks characterized by high input dimensionality requires suitable approximations because of the prohibitive demands on computation and samples. An algorithm is proposed that is based on a "greedy" selection of the features and that takes both the *mutual information* with respect to the output class and with respect to the already-selected features into account. Finally the results of a series of experiments are discussed.**

Index Terms—**Feature extraction, neural network pruning, dimensionality reduction, mutual information, supervised learning, adaptive classifiers.**

I. Introduction

DURING the development of neural net classifiers the "preprocessing" stage, where an appropriate number of relevant features is extracted from the raw data, has a crucial impact both on the complexity of the learning phase and on the achievable generalization performance. While it is essential that the information contained in the input vector is sufficient to determine the output class, the presence of too many input features can burden the training process and can produce a neural network with more connection weights that those required by the problem.

From an application-oriented point of view, an excessive input dimensionality implies lengthened preprocessing and recognition times, even if the learning and recognition performance is satisfactory.

In this paper we consider the use of the *mutual information* (MI for short) to evaluate the "information content" of each individual feature with regard to the output class. The approximated evaluation of the *mutual information* of each candidate feature is the starting component of a "pruning" algorithm that selects a subset of relevant features from an initial set of available features. In addition to their practical use for limiting the input dimensionality, the analysis based on the *mutual information* provides the developer with a useful diagnosis of the relevance of different features and of the mutual dependencies.

Different feature selection methods have been analyzed in the past. For example, in [10] the irrelevant features are eliminated as a consequence of a pruning of the weights, that considers the *sensitivity* of the global error function E to the presence or absence of the different synapses. The *sensitivity* is estimated by integrating the partial derivatives $\partial E/\partial w$ on the path in weight space traced during the learning process. Our method, while producing similar results in test cases, is applied before learning starts and therefore does not depend on the learning process. Other techniques are based on *linear* transformations of the input vector. In [14] the Karhunen-Loe've transformation is applied so that the transformed coordinates can be arranged in order of their "significance," considering first the components corresponding to the major eigenvectors of the correlation matrix. In [18] different feature evaluation methods are compared. In particular the method based on *principal component analysis* (PCA) evaluates the features according to the projection of the largest eigenvector of the correlation matrix on the initial dimensions, the method based on Fisher's linear discriminant analysis evaluates them according to the magnitude of the components of the discriminant vector.

A major weakness of these methods is that they are not invariant under a transformation of the variables. For example a linear scaling of the input variables (that may be caused by a change of units for the measurements) is sufficient to modify the PCA results. Feature selection methods that are sufficient for simple distributions of the patterns belonging to different classes can fail in classification tasks with complex decision boundaries. In addition, methods based on a linear dependence (like the *correlation*) cannot take care of arbitrary relations between the pattern coordinates and the different classes. On the contrary, the *mutual information* can measure arbitrary relations between variables and it does not depend on transformations acting on the different variables.

In the following sections, first we summarize the relevant concepts (Section II), then we analyze the practical applicability of the *mutual information* and propose an approximated algorithm with a low computational complexity and with limited requirements on time and on the number of training examples (Section III). Finally we present some experimental

Manuscript received June 17, 1992; revised September 16, 1992.

The author is with the Dipartimento di Matematica, Università di Trento, 38050 Povo (Trento), Italy.

IEEE Log Number 9205932.

results of our algorithm for a series of classification problems (Section IV).

II. Background

A. Definition of the Mutual Information

An operating classifier (consider for example a multilayer perceptron trained to classify patterns from a set of different classes with the backpropagation algorithm described in [19]) can be considered as a system that reduces the initial uncertainty, to be defined precisely later, by "consuming" the information contained in the input vector. In the ideal case the final uncertainty will be zero (i.e., the class will be certain), in actual "real world" applications the final uncertainty can be higher for at least two different reasons, insufficient input information or suboptimal operation. In the second case the available information can be sufficient to resolve all ambiguities but the network "wastes" some of it because of insufficient training, approximations or failures. While this case can be remedied by considering additional training examples, a longer training period or different algorithms, the lack of sufficient information should be detected as soon as possible in the development process because in this case the only remedy is that of adding more features or considering more informative ones.

Shannon's information theory (see [20]) provides a suitable formalism for quantifying the above concepts. If the probabilities[1] for the different classes are $P(c); c = 1, \dots N_c$, the initial uncertainty in the output class is measured by the *entropy*:

$$H(C) = -\sum_{c=1}^{N_c} P(c) \log P(c) \tag{1}$$

while the average uncertainty after knowing the feature vector $\boldsymbol{f}$ (with N_f components) is the *conditional entropy*:

$$H(C|F) = -\sum_{f=1}^{N_f} P(\boldsymbol{f}) \left(\sum_{c=1}^{N_c} P(c|\boldsymbol{f}) \log P(c|\boldsymbol{f}) \right) \tag{2}$$

where $P(c|\boldsymbol{f})$ is the conditional probability for class c given the input vector $\boldsymbol{f}$. If the feature vector is composed of continuous variables, the sum will be replaced by an integral and the probabilities by the corresponding probability densities. For example, in one dimension, one has:

$$H(F) = -\int P(f) \log P(f)\, df \tag{3}$$

Note that the entropies of continuous systems *depend* on coordinates. For a linear transformation with $f \to f' = \alpha f$, the above integral becomes

$$\begin{aligned} H'(F) &= -\int P'(f') \log P'(f') df' \\ &= -\int P(f) \log \left(\frac{P(f)}{\alpha} \right) df = H(F) + \log \alpha \end{aligned} \tag{4}$$

[1] About the notation: for simplicity we indicate the different probability densities with the same $P()$ function. Its meaning is easily derived from the variable contained. For example $P(c)$ is the value of the density function for the "class" variable (i.e., $P_c(c)$), $P(f)$ for the "feature" variable (i.e., $P_f(f)$).

In general, the conditional entropy will be less than or equal to the initial entropy. It is equal if and only if one has independence between features and output class (i.e., if the *joint* probability density is the product of the individual densities: $P(c, \boldsymbol{f}) = P(c)P(\boldsymbol{f})$). The amount by which the uncertainty is decreased is, by definition, the *mutual information* $I(C; F)$ between variables c and $\boldsymbol{f}$:

$$I(C; F) = H(C) - H(C|F) \tag{5}$$

This function is symmetric with respect to C and F and, with simple algebraic manipulations, can be reduced to the following expression:

$$I(C; F) = I(F; C) = \sum_{c,f} P(c, \boldsymbol{f}) \log \frac{P(c, \boldsymbol{f})}{P(c)P(\boldsymbol{f})} \tag{6}$$

The *mutual information* is therefore the amount by which the knowledge provided by the feature vector decreases the uncertainty about the class. If one considers the uncertainty in the combined events $(c, \boldsymbol{f})$, i.e., $H(C; F)$, in general this is less than the sum of the individual uncertainties $H(C)$ and $H(F)$ and it is possible to demonstrate the following relation:

$$H(C; F) = H(C) + H(F) - I(C; F) \tag{7}$$

The combined uncertainty is reduced because of the information that one variable provides about the other one. If the feature vector has continuous components, one obtains:

$$I(C; F) = I(F; C) = \sum_{c} \int P(c, \boldsymbol{f}) \log \frac{P(c, \boldsymbol{f})}{P(c)P(\boldsymbol{f})} d\boldsymbol{f} \tag{8}$$

The argument of the logarithm in (8) is now dimensionless, so that the MI does *not* depend on a transformation of variables[2]. The MI is a function of the *joint* probability distribution of the two variables c and $\boldsymbol{f}$. For a qualitative explanation, let's consider a particular value of $\boldsymbol{f}$. The contribution to the integral is large and positive if the distribution $P(c, \boldsymbol{f})$ is "uneven" (at the limit peaked for a single c, the "correct" class), and tends to zero for the limit of a flat distribution for c, given by $P(c)P(\boldsymbol{f})$. The MI measures the "lumpiness" of the joint distribution.

Although the main motivation of Information Theory was the engeneering of "noisy" communication channels, its concepts have been applied to different fields, in particular [9] considers the implications for statistical decisionmaking, a field closely related to pattern recognition and classification, [6] uses the *mutual information* to find the optimal time delay to construct a multidimensional *phase portrait* of a dynamical system, with implications for the prediction of temporal series. In the field of neural networks, methods and concepts from Information Theory have been used, for example, in [13] for the generation of ordered maps. Training algorithms based on the MI are considered in [2], where the training criterion is based on the relative entropy (i.e., the likelihood of the targets given the networks outputs), in [1] where the minimization of the conditional class entropy is the basis of a learning algorithm that builds a multilayer network, and in [23] for one case of unsupervised learning.

[2] The transformations considered are invertible and differentiable.

B. Advantages over Correlation

It is well known that the main advantage of the multilayer perceptron over the simple perceptron model is given by its capability of realizing arbitrary continuous mappings between inputs and outputs [7]. For classification, this result implies that a multilayer perceptron with at least one hidden layer can realize arbitrary *nonlinear separations* between different classes[3].

While linear methods of analysis (like the *correlation*) can be useful in particular cases, in general it is essential to consider also nonlinear relations between different variables. The motivation for considering the MI is its capability to measure a general dependence between two variables.

For example, to realize the classification given by the exclusive OR function of two input variables (with equal probabilities for the possible inputs), the correlation Γ between any input variable x and the output variable y is zero ($\Gamma = \sum_i \sum_j P_{ij} x_i y_j - (\sum_i P_i x_i)(\sum_j P_j y_j) = 0$), while the MI between the input vector and the output is $\log_2 2$ *bits*, equal to the initial uncertainty of one *bit*[4]: the input vector determines the output class with no ambiguity. In other words, two variables x and y are *linearly independent* if $E(xy) = E(x)E(y)$ (E being the expectation) and *generally independent* if $P(x,y) = P(x)P(y)$. General independence implies linear independence, but not vice versa. While the difference between MI and correlation for Gaussian random variables is trivial (in this case from the correlation $C_{xy} \equiv \sigma_{xy}/\sqrt{\sigma_x \sigma_y}$, where σ_i is the standard deviation and σ_{ij} the covariance matrix, one can derive the MI as $I(X,Y) = -(1/2)\log[1 - C_{xy}^2]$, see [5]) and two variables are linearly independent if and only if they are generally independent, for complex probability densities the concept of linear dependence is not a very useful one. A detailed investigation of the advantages of the MI versus the correlation is contained in [5] and [12].

III. Selecting Features with the Mutual Information

In the development of a classifier one often is confronted with practical constraints on the hardware and on the time that is allotted to the task. While many kinds of features can be extracted from the raw data (consider for example an Optical Character Recognition task) and the information contained in them is sufficient to determine the class with low ambiguity, one may be forced to reduce an initial set of n features to a smaller set of k features, where the number k is related to the practical constraints. Let's abstract from the above considerations the following "feature reduction" problem:

[FRn-k:] Given an initial set of n features, find the subset with $k < n$ features that is "maximally informative" about the class.

[3]One can map patterns of the ith class to an output activation vector with value 1 in the ith place, and 0 otherwise. Continuity of the mapping can be obtained by a thin transition region on the boundaries between different classes.

[4]In fact, the probability $P(\boldsymbol{f} = (f_1, f_2), y)$ is different from zero only when $y = XOR(f_1, f_2)$. In this cases $P((f_1, f_2), XOR(f_1, f_2)) = 1/4$ and the argument of the logarithm is $2 = (1/4)/(1/4\,1/2)$.

In the framework of Information Theory, remembering (5) and the fact that the class uncertainty is fixed, the problem can be reformulated as follows:

[FRn-k] Given an initial set F with n features, find the subset $S \subset F$ with k features that minimizes $H(C|S)$, i.e., that maximizes the *mutual information* $I(C;S)$.

Unfortunately, the practical applicability of the above solution to complex classification problems requiring a large number of features is limited because of two computational problems. First the number of samples and the amount of CPU time required for computing the MI become prohibitive when the dimensionality of the feature vector $\boldsymbol{f}$ is large. For example Fraser's method (see [6]), that is a computationally efficient algorithm for calculating the MI, requires for its convergence a number of samples "in the millions" when the number of features in the input vector is larger than 3 or 4, clearly an exorbitant number for "real world" classifier development. Even assuming that a suitable example set can be constructed, the consideration of *all* possible subsets requires a number of runs equal to $\binom{n}{k}$.

One is therefore forced to consider approximated solutions of the **FRn-k** problem. An approximated solution is acceptable also because there is no guarantee that the optimal subset of features will be processed in the optimal way by the learning algorithm and by the operating classifier. Although it is necessary, the availability of an "informative" input vector is not sufficient for the development of a correct classifier.

A. Our Algorithm (MIFS)

Motivated by the above reasons, we considered two approximations for the **FRn-k** problem. First the MI between vector variables is approximated using the MI between the individual components of the vectors. Instead of calculating the *mutual information* $I(F;C)$ between a feature *vector* $\boldsymbol{f}$ and the class variable c we compute only $I(f,C)$ and $I(f,f')$ where f and f' are *individual* features. In this case the "computationally impossible" calculation of the exact MI is substituted with a series of feasible calculations. Then the analysis of all possible subsets is substituted by a "greedy" algorithm. Given a set of already selected features, the algorithm chooses the next feature as the one that maximizes the information about the class corrected by subtracting a quantity proportional to the average MI with the selected features. In order to be selected, a feature must be informative about the class without being predictable from the current set of features. For example, if two features f and f' are highly dependent, $I(f,f')$ will be large and, after the better one is picked, the selection of the second one is penalized.

The MIFS algorithm ("mutual information based feature selection") can be described by the following procedure:

1) (Initialization) Set $F \leftarrow$ "initial set of n features;" $S \leftarrow$ "empty set."
2) (Computation of the MI with the output class) for each feature $f \in F$ compute $I(C;f)$.
3) (Choice of the first feature) find the feature f that maximizes $I(C;f)$; set $F \leftarrow F \backslash \{f\}$; set $S \leftarrow \{f\}$

4) (Greedy selection) repeat until $|S| = k$:
   a) (Computation of the MI between variables) for all couples of variables (f, s) with $f \in F$, $s \in S$ compute $I(f; s)$, if it is not already available.
   b) (Selection of the next feature) choose feature f as the one that maximizes $I(C; f) - \beta \sum_{s \in S} I(f; s)$; set $F \leftarrow F \setminus \{f\}$; set $S \leftarrow S \cup \{f\}$
5) Output the set S containing the selected features.

The parameter β regulates the relative importance of the MI between the candidate feature and the already-selected features with respect to the MI with the output class. If β is zero, only the MI with the output class is considered for each feature selection. If β increases, this measure is discounted by a quantity proportional to the total MI with respect to the already-selected features. In practice, we find that a value for β between 0.5 and 1 is appropriate for many classification tasks (these are the values used for the tests in Section IV).

At this point it is important to remark that, while the use of the MI between features and output class to rank the relevance of each isolated component is theoretically justified, the summation of the "two-point" MI's to consider the dependencies between different features during the selection process is an heuristic approximation whose effectiveness must be tested in the field for the different classification problems.

B. Estimation of the MI from Samples

Because the MI is calculated by estimating the probability density from a finite number of samples, we must check that the errors caused by the estimation do not impair the above selection process.

Let's assume that we have a number N of examples in the training set and that the probability densities $P(c)$, $P(f)$ and $P(c, f)$ are approximated by histograms, i.e., by counting the number of cases with values of the variables belonging to a set of intervals ($P_c = n_c/N$, $P_f = n_f/N$, $P_{cf} = n_{cf}/N$, where n is the number of occurrences for the given interval). Finally, let K_f be the number of intervals for the f variable and K_c the number of intervals for the class variable, i.e., the number of classes.

By adapting to our case the analysis of [12], the difference between the true value $\overline{I}$ and the estimation I of the *mutual information* can be approximated as follows:

$$\Delta I \equiv I - \overline{I} \approx \frac{1}{2N}\left(\sum_{c,f}\frac{(\delta n_{cf})^2}{n_{cf}} - \sum_{c}\frac{(\delta n_c)^2}{n_c} - \sum_{f}\frac{(\delta n_f)^2}{n_f}\right) \tag{9}$$

where the sums are over the discretization intervals and δn are the fluctuations of the countings with respect to the mean values ($\delta n = n - \overline{n}$). The approximation is valid up to the second order of the relative fluctuations and if the ratios $\overline{n}_{cf}/\overline{n}_c\overline{n}_f$ do not change very much with c and f (see [12] for the details). Now, because the typical fluctuation of the countings is of the order of the square root of the mean values, we can arrive at the following approximation:

$$\Delta I \approx \frac{1}{2N}(K_c K_f - K_c - K_f) \tag{10}$$

TABLE I
OVERESTIMATION OF THE MI AND COMPARISON WITH THE ESTIMATED ERROR $\Delta I = 4/N$

number of examples	$I(x; class)$	ΔI	$I(y; class)$	ΔI	$\Delta I = 4/N$
10	0.800	0.468	0.315	0.182	0.4
100	0.513	0.045	0.183	0.050	0.04
1000	0.491	0.023	0.147	0.014	0.004
10000	0.468	*	0.133	*	*

Note that, in this approximation, the MI is overestimated (in practical cases $K_c K_f - K_c - K_f > 0$) and this overestimation depends only on the number of quantization levels. The fact that the MI is overestimated in the same way for the different variables limits the estimation effects on the relative *ranking* of the different features[5], and therefore the effects on the MIFS algorithm. The number of quantization levels K_f has to be appropriately chosen. If the statistical distributions have a lot of structure, using a small number of levels will cancel these details and reduce the estimated MI. But using too many levels K_f will produce the estimation problems previously described. In practice, we obtained good results by using $K_f = 10$ levels and cutting the range of values into equal-sized intervals[6].

In Table I we present the results of an experiment for a two-class discrimination problem. Patterns in two dimensions are generated with equal probability for the two classes, where one is described by a central Gaussian distribution and the other by two lateral Gaussians. With the notation that will be introduced in Section IV (see (18)), the two densities are

$$p_1(x, y) = N(x, 0, 0.2)N(y, 0, 0.4);\ p_2(x, y) = N(y, 0, 0.2)(N(x, -0.5, 0.2) + N(x, 0.5, 0.2))/2$$

In this case $N_c = 2$, $N_f = 10$, $K_f = 10$, and the approximation in (10) is acceptable (the differences ΔI are calculated with respect to the "true" value calculated for $n = 10000$).

An algorithm for calculating the MI from samples that is based on an adaptive discretization (i.e., a variable size of the intervals so that a sufficient number of samples is contained in each of them) is presented in [6]. Fraser's algorithm is based on the invariance of the MI with respect to transformations acting on the individual coordinates and on a recursive sequence of partitions of the space of the variables. Each recursive call goes deeper in areas where the joint distribution has finer structure and is terminated when the number of samples in an element of the partition becomes insufficient for an accurate evaluation. The computational complexity of the algorithm is $N \log N$ for a number of samples equal to N, and therefore it permits a fast evaluation also for large numbers of examples. Fraser's algorithm has been used in the Optical Character Recognition tests described in Section IV-E. For the reader's convenience, a short description of the algorithm is provided in the Appendix.

[5] Let us suppose that features a and b have "true" mutual informations with respect to the output $I_a > I_b$, in the approximation of equation 10 we will still have $(I_a + \Delta I) > (I_b + \Delta I)$ for the estimated quantities.

[6] If the distribution for the values of one variable is not known *a priori*, we calculate its mean μ and standard deviation σ, and cut the interval $[\mu - 2\sigma, \mu + 2\sigma]$ into K_f equal segments. The rare points falling outside are assigned to the extreme left (or right) segment when histograms are calculated.

IV. Experimental Results

A. Simple Test Cases

We show here the results for two test cases derived from [10].

Example 1: The first classification problem is illustrated in Fig. 1. The feature vector (X, Y) is uniformly distributed in $[0,1] \times [0,1]$, one pattern belongs to class "1" if the two inequalities $x < \alpha$ and $y < \beta$ ($\beta = 1/(2\alpha)$), to class "2" otherwise. By calculating the MI between each feature and the class, one obtains the following result:

$$I(X;C) = 1 + \alpha \log\left(\frac{2\alpha - 1}{2\alpha}\right) - \frac{1}{2}\log(2\alpha - 1) \qquad (11)$$

$I(Y;C)$ is obtained from (11) by substituting α with $\beta = 1/(2\alpha)$. From (11) one derives that feature X is more informative than y as long as $1/2 \leq \alpha < 1/\sqrt{2}$. Because the better feature is selected before the learning process is started, the choice does not depend on the details of the learning algorithm (like the initial weight values and a proper convergence).

In this case the same choice of the most informative feature is obtained by using the Fisher linear discriminant vector. The *Fisher linear discriminant* is defined as that linear function $y = \boldsymbol{w}^t \boldsymbol{x}$ for which the *criterion function*

$$J(\boldsymbol{w}) = \frac{|\tilde{m}_1 - \tilde{m}_2|^2}{\tilde{s}_1^{\,2} + \tilde{s}_2^{\,2}} \qquad (12)$$

is maximum, where $\tilde{m}_i$ is the sample mean for the projected points ($\tilde{m}_i = (1/n_i)\sum_{y \in class_i} y$) and $\tilde{s}_i$ the *scatter* for the projected samples ($\tilde{s}_i = \sum_{y \in class_i}(y - \tilde{m}_i)^2$). The task is that of maximizing the ratio of between-class to within-class scatter. The difference of the projected means has to be large relative to a measure of the standard deviation for each class. The solution (see [3]) is:

$$\boldsymbol{w} = S_W^{-1}(\boldsymbol{m}_1 - \boldsymbol{m}_2) \qquad (13)$$

where $\boldsymbol{m}_i$ is the d-dimansional sample mean for class i and S_W is the sum of the two scatter matrices S_i defined as follows

$$S_i = \sum_{\boldsymbol{x} \in class_i} (\boldsymbol{x} - \boldsymbol{m}_i)(\boldsymbol{x} - \boldsymbol{m}_i)^t \qquad (14)$$

For the above classification problem the expected *scatter matrices* S_i for N sample points are given by (15) and (16).

If we rate the "importance" of the ith feature according to the ith component of the Fisher vector, the more informative feature is x if the value of the parameter α is between $1/2$ and $1/\sqrt{2}$ and y for larger values, as it was the case by using the MI. In Fig. 1 we compare the graphs of the magnitude of the x and y components of the normalized Fisher vector and of the value of the MI for the x and y coordinates.

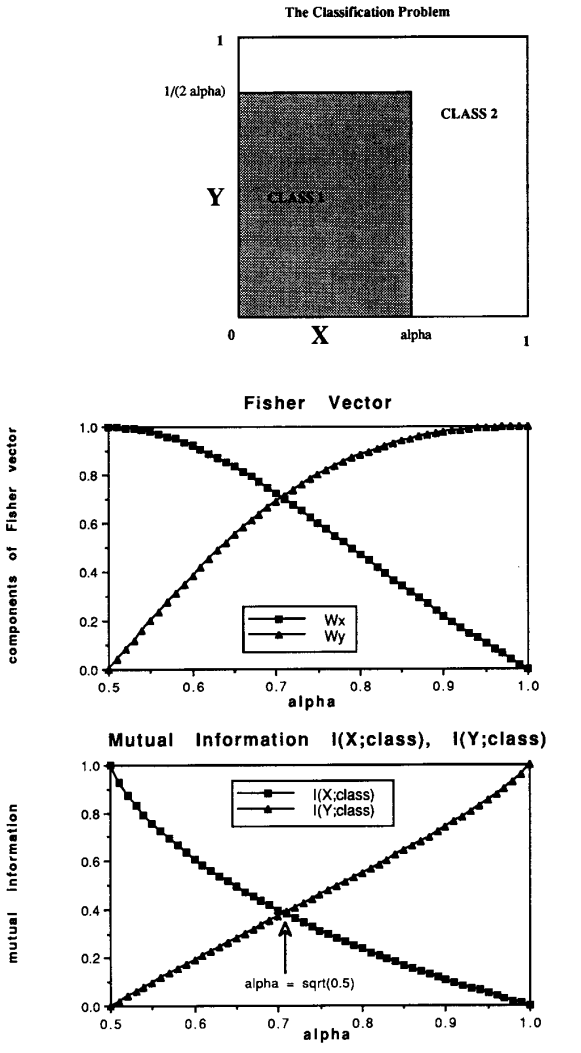


Fig. 1. Comparison of *mutual information* and Fisher's linear discriminant analysis for feature selection. The two-class discrimination problem is illustrated at the top. The components of the normalized Fisher vector and the MI below. Both methods select x as the more informative feature if α is less than $1/\sqrt{2}$, y in the other case.

Note that the patterns are scattered in the same way along the X and Y coordinates, so that the Principal Component Analysis (whose result does not depend on α) does not help in choosing the most appropriate feature.

Example 2: This example (the "rule-plus-exception" problem) is derived from [16] and used in [10]. The classification problem on an input space with four binary variables is defined

$$S_1 = \begin{pmatrix} \frac{N}{24}\alpha^2 & 0 \\ 0 & \frac{N}{96\alpha^2} \end{pmatrix} \qquad (15)$$

$$S_2 = \begin{pmatrix} \frac{N}{48\alpha}(2\alpha^4 - (3\alpha - 2)^3 + (2\alpha - 1)(2 - \alpha)^3) & -\frac{N}{8\alpha}(3\alpha - 1 - 2\alpha^2) \\ 0 & \frac{N}{192\alpha^3}(1 - \alpha(3 - 4\alpha)^3 + (1 - \alpha)(4\alpha - 1)^3) \end{pmatrix} \qquad (16)$$

by the Boolean function $AB + \overline{A}\ \overline{B}\ \overline{C}\ \overline{D}$. The output class is "true" when the "rule" AB is true or when the "exception" $\overline{A}\ \overline{B}\ \overline{C}\ \overline{D}$ occurs. Clearly the "rule" is more important than the "exception" because it accounts for 15 out of 16 correct decisions and therefore the relevant variables are A and B. This is confirmed by calculating the *mutual information*. One obtains $I(A;OUT) = I(B;OUT) = 0.124$, $I(C;OUT) = I(D;OUT) = 0.013$. Again the fact that variables A and B are more relevant can be detected from the beginning, thereby "pruning" the network before learning is started.

B. Mixture of Gaussian Densities

In a mixture of Gaussian densities the samples are assumed to be generated by selecting a "prototype" c_i with probability $P(c_i)$ and then selecting a pattern $\boldsymbol{x}$ with a normal (Gaussian) probability $P(\boldsymbol{x}|c_i)$. A general multivariate normal density in d dimensions can be written as:

$$p(\boldsymbol{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{m})^t\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{m})] \quad (17)$$

where $\boldsymbol{m}$ is the *mean vector* ($\boldsymbol{m} = E[\boldsymbol{x}]$) and Σ is the *covariance matrix* ($\Sigma = E[(\boldsymbol{x}-\boldsymbol{m})(\boldsymbol{x}-\boldsymbol{m})^t]$). We now present the results of some two-class discrimination experiments, where each class is described by a simple mixture of Gaussian densities, showing the *robustness* of the MI criterion with respect to different distributions of the class densities.

Let's consider an input space with $N_f = 2$ features and two categories, where the first one is described by a Gaussian distribution with zero mean that is progressively elongated along the x dimension in the different tests (by increasing σ_{1x}), and the second one is a normal distribution with a fixed standard deviation that is displaced in the x direction with respect to the first one. After introducing the one dimensional distribution

$$N(v,\mu,\sigma) = (2\pi\sigma^2)^{-1/2} \exp\left[-\frac{(v-\mu)^2}{2\sigma^2}\right] \quad (18)$$

that is a Gaussian with mean μ and variance σ^2, the probability densities for classes 1 and 2 are

$$p_1(x,y) = N(x,0,\sigma_{1x})N(y,0,0.1);$$
$$p_2(x,y) = N(x,0.5,0.1)N(y,0,0.1)$$

To simulate a real classification task, we extracted 1000 patterns with equal probability from the two distributions for each of a series of tests with increasing values of σ_{1x}, as illustrated in Fig. 2 (for the cases with σ_{1x} from 0.1 to 0.8). From the classes' definition, it is apparent that the y component of the pattern is completely useless because the patterns are distributed in the same way for the two classes along the y coordinate, while the x component is sufficient to determine the class with a low degree of error.

The approximated MI between the two input variables and the class ($I(x;class)$ and $I(y;class)$) are listed in Table II. As expected, the MI is close to zero for the y variable and significant for the x variable. In fact, it is close to 1 (the output uncertainty) if the two classes are well separated ($\sigma_{1x} = 0.1$), it decreases when the first class "expands" and covers the second one, and increases again for large values of σ_{1x}. In this last case the probability that the x coordinate of a pattern belonging to class "two" falls in the region of class "one" becomes small and smaller. For comparison, the results of Fisher's linear discriminant analysis are listed in the second and third columns of Table II. For large values of σ_{1x} the magnitude of the components of the Fisher vector is not related to the discrimination capability of the two coordinates. For example, for $\sigma_{1x} = 3.2$ the more informative feature appears to be the second one. This is due to the increasing spread of class "one" along the x dimension: although the difference of the means for the two classes has a y component equal to zero (and therefore the *criterion function* (12) is zero for a vector along the y direction), the difference *estimated* from the finite number of samples has a small (random) y component that is causing the misleading result. In addition, the linear discriminant analysis is not defined if the classes have the same mean and it encounters serious estimation problems for small values of the *between-class* scatter, measured by the difference between the means. If this is small with respect to the standard deviations of the classes the results will depend on random fluctuations.

TABLE II
COMPONENTS OF NORMALIZEDFISHER VECTOR AND MUTUAL INFORMATION

σ_{1x}	W_x	W_y	$I(x;class)$	$I(y;class)$
0.1	0.999	−0.032	0.964	0.001
0.2	0.999	−0.028	0.792	0.001
0.4	0.999	−0.012	0.539	0.001
0.8	0.998	0.048	0.612	0.001
1.6	0.952	0.304	0.692	0.001
3.2	0.489	0.871	0.679	0.001
6.4	0.049	0.998	0.728	0.001

The above considerations can be extended to the n-dimensional case (the two-dimensional case was chosen only for display purposes) and to the mixture of different distributions.

C. Classification of Sonar Targets

The task is to train a network to discriminate between sonar returns bounced off a metal cylinder and those bounced off a roughly cylindrical rock. The data set has been used in [8], where a multilayer neural network is trained for the classification[7]. The purpose of the following tests is that of comparing the relative advantages of different techniques for dimensionality reduction. Our training and testing sets refer to the "aspect angle dependent" series of experiments in [8]: the 104 training and 104 testing patterns are selected to include all target aspect angle.

Mutual Information Diagram and Feature Selection: The original sonar signal is filtered, Fourier transformed and a set of 60 features is extracted by integrating the spectrogram over sampling apertures with varying temporal offsets (to correspond to the slope of the FM chirp).

[7]The data set was obtained from the "neural net benchmark collection" organized by Scott E. Fahlman at the Carnegie Mellon University. It was contributed by Terry Sejnowski, now at the Salk Institute and the University of California at San Diego, who developed it in collaboration with R. Paul Gorman of Allied-Signal Aerospace Technology Center.

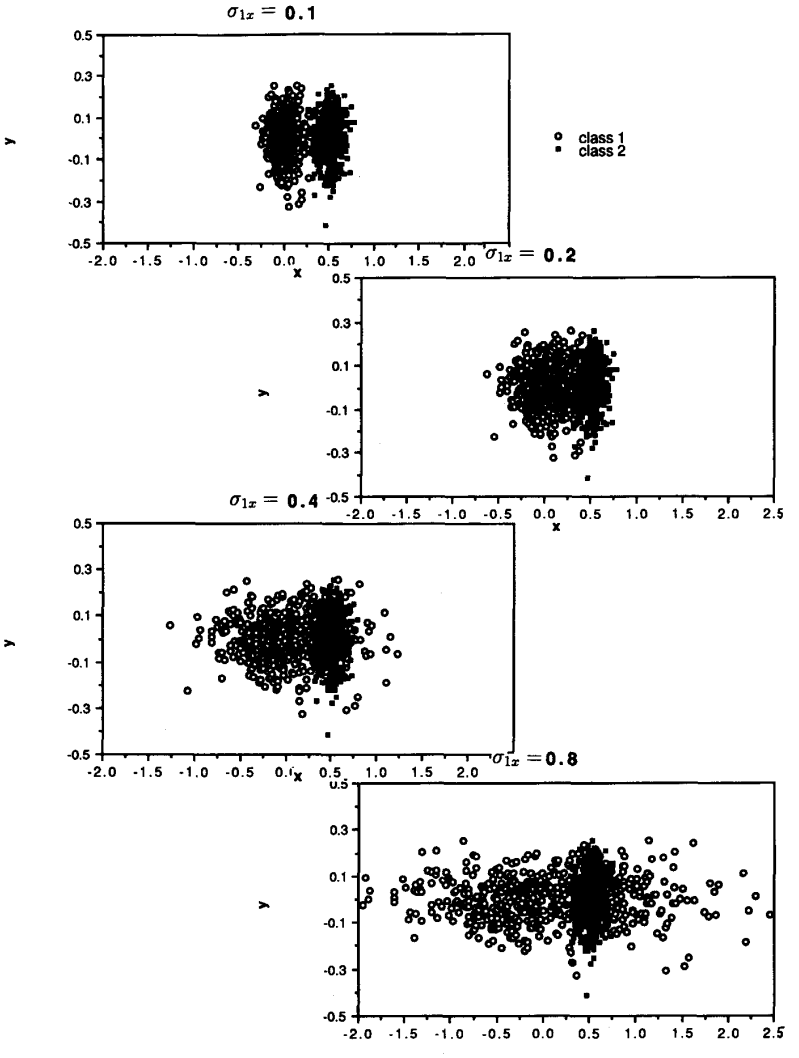


Fig. 2. Discrimination task with Gaussian densities. The standard deviation along the x axis of the distribution for class 1 is increasing from the top (σ_{1x} = 0.1) to the bottom (σ_{1x} = 0.8).

In Fig. 3 we show the *mutual information diagram* of the signal, i.e., the value of the MI between the different features and the output class. The MI diagram provides useful information to the developer of a classification system. In this case there are peaks in the MI for the region corresponding roughly to the "attack" and "decay" features of [8], although we did not investigate the possible correlations with human perceptual cues. It is also apparent that some features have a very low MI. The developer can use the MI diagram to diagnose the feature extraction phase, for example to eliminate some features that have a very low information content.

A different type of diagnosis is provided by the *MI function*, of the *mutual information* between each feature and the other ones, as a function of a parameter describing the relative feature location (in this case the parameter is given by the relative times at which the different sampling apertures are positioned). The *MI function* can be compared to the more traditional *correlation function*, with the difference that the *MI function* measures a general dependence between variables, in comparison with a linear dependence. In addition, the *MI function* can be applied equally well to numerical and *symbolic* sequences, like the sequence of letters in a text (see [12]).

In Fig. 4(a) we show the *MI function* for one particular feature (feature 8) with respect to the other ones. One can identify a peak that is decaying for near features (this result is related to the temporal superposition of the different sampling windows and to the dependence between the characteristics of the signal at contiguous times) and a more complex structure for "distant" features. This behavior is qualitatively similar for the other features. In Fig. 4(b) the *MI function* is shown in more detail for the first feature. Again the MI decays gradually for features corresponding to later times until a plateau with a complex structure is reached. The *MI function* can be used to identify relations between different features. If some features

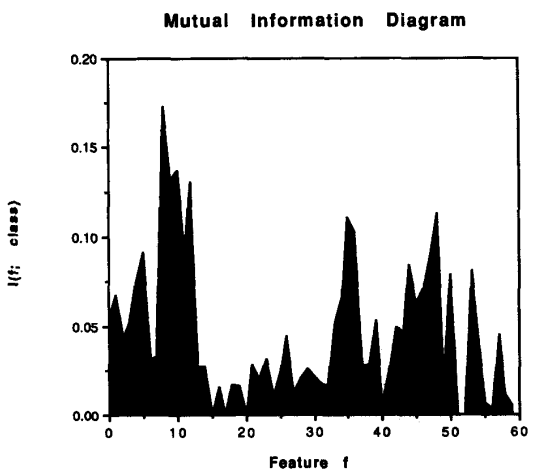


Fig. 3. Mutual information diagram for the features extracted from the sonar signal.

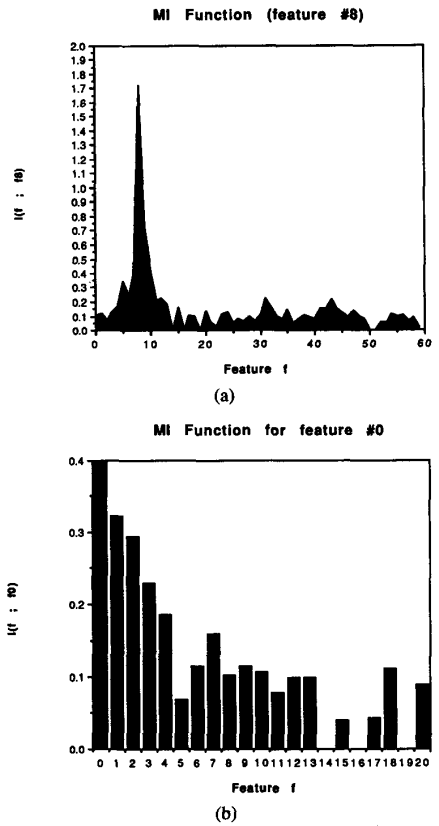


Fig. 4. Mutual information function for the sonar classification task. The MI between feature 8 and the others is shown in (a), the MI between feature 0 and the nearest features in (b).

are highly dependent it is possible that some of them are redundant and can be eliminated.

It is interesting to compare the selection order given by the MIFS algorithm presented in Section III (with $\beta = 1$) and the ranking scheme based only on the values of the MI with respect to the output class (i.e., $\beta = 0$). In the first case, after the feature with the highest MI is selected, the choice tends to jump to distant places because the subsequent features are chosen by taking into account both the MI with respect to the class and the MI with the already-selected features. In the second case, all features in a peak of the MI diagram are picked before the other candidates are considered (see Fig. 5, where each line specifies the selected features, with their number k growing from the top to the bottom). For example, if four features are selected, in the first case all four come from the tallest peak and are extracted from a small time interval, in the second case they correspond to sampling apertures spread over the entire signal.

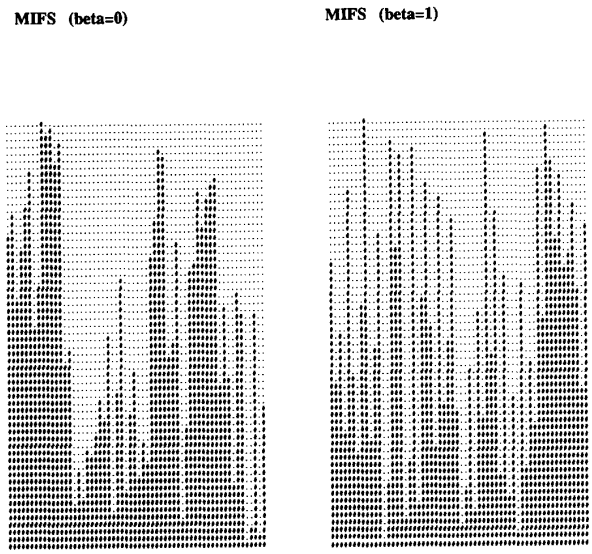


Fig. 5. Selection order by the MIFS algorithm with $\beta = 0$ (left) and $\beta = 1$ (right). The number k of selected features increases from the top ($k = 1$) to the bottom ($k = 60$). The line for a given k shows the chosen features (with "#").

Learning and Generalization for Different Pruning Techniques: We consider here the effect of different dimensionality reduction techniques on the performance of a multilayer perceptron neural network trained for the sonar classification problem. The training algorithm and parameters are the same as those used in [8], the network architecture has an input layer of variable size (corresponding to the dimension of the reduced pattern), one hidden layer with three units, and two output units coding for the two classes. In three series of experiments, the input vector is reduced to 10%, 20%, and 30% of its original size and, for each size, a set of 10 runs is executed (by varying the seed of a random number generator used for initializing the weights and — in the case of a random pruning — for selecting the features). We then calculate the average performance and its standard deviation.

The training curves (percent classification as a function of the number of *on-line* pattern presentations) are similar to those of [8]. In Fig. 6 we show an example of a learning stage (for the 10% cut) and the average on 10 tests.

The following results refer to the generalization performance of the networks (measured on the disjointed test set) as

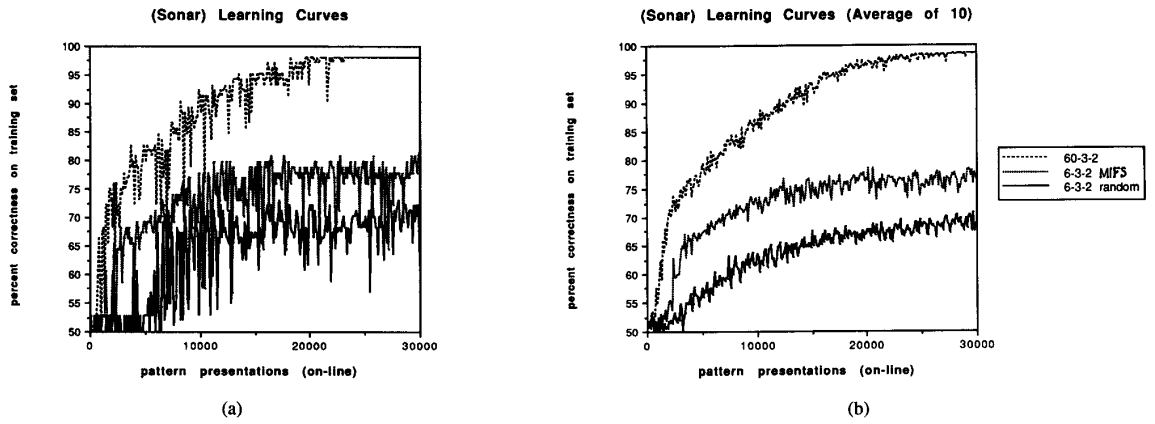


Fig. 6. Training curves in the sonar problem, for the original architecture (60-3-2) and for the reduced net (6-3-2). Single run (above) and average of ten (below).

TABLE III
COMPARISON OF FEATURE SELECTION TECHNIQUES FOR THE SONAR PROBLEM

number of features	method	performance on test set	standard deviation
6	MIFS	75.1	4.1
	random	68.1	4.1
	PCA	63.2	4.0
12	MIFS	78.9	3.0
	random	76.9	4.1
	PCA	63.7	3.2
18	MIFS	79.2	1.3
	random	78.5	5.1
	PCA	72.7	3.7

a function of the number of iterations. Learning is executed for 120 000 *on-line* pattern presentations. The training period is increased with respect to [8] because the *over-training* phenomenon (i.e., a decrease in generalization performance because of an excessive training causing the "memorization" of the training set) is quite difficult to observe in this particular case, if it is present at all, and we wanted to be reasonably sure that the net reached the maximum generalization performance. The three methods that we compare are the MIFS algorithm in Section III, the scheme based on the Principal Component Analysis (PCA) (see [18]) and, finally, a random dimensionality reduction (see Fig. 7).

The performance of the networks at the end of the training period is listed in Table III. The performance of the original network (60 inputs) with the architecture 60-3-2 is 86.5% (standard deviation 3.0).

In this case the superiority of the MIFS technique emerges more clearly for significant reductions of the number of features (e.g., when they are reduced from 60 to 6–12), while the difference with respect to a random reduction tends to decrease for smaller reductions (although the standard deviation for the random reduction is larger). This is to be expected for this particular problem where there is a high degree of dependence between features extracted from near time intervals of the signals: if a large fraction of the original set of features is maintained and if these are "spread" over the entire signal duration, the information loss with respect to the amount contained in the original signal will be very small and will not depend on the selection method in a crucial manner.

The PCA method of [18] is not to be confused with the use of the Karhunen-Loe've transformation in [14]. In our case we are not considering feature transformations but only the selection of a subset of optimal features from a given vector.

D. The Iris Data

The data were listed and used by R. A. Fisher in his classic paper on discriminant analysis [4]. They are from measurements by E. Anderson on 150 samples of three species of iris[8]. The input pattern is composed of four features[9].

In this case (given the limited number of features) we reduce the input vector by 50% and present the results for all possible (six) selections of two features. A multilayer perceptron with the architecture 2-4-3 is trained on a subset of 100 cases and tested on the remaining 50 cases. The learning rate for the *on-line* backpropagation algorithm is 0.002 (no momentum) and weights are randomly initialized in the range $[-0.5, 0.5]$. The results are an average on ten runs.

In Fig. 8 we present the generalization results for all subsets of two features (indicated by a binary number, where "1" means that the corresponding feature is present). It is manifest that there are two optimal subsets ("1001" and "1010") with a correctness of approximately 93%, three suboptimal selections with performance in the region 80–90% and a bad selection ("1100") that reaches only 60%. The standard deviation is approximately 2.0 for the case "0011" and 1.0 for all other cases. The MIFS algorithm chooses one of the two optimal sets (precisely the set composed of features 1 and 3).

[8] The data were obtained from Russel Leighton at MITRE Signal Processing Center. They are in the examples that come with the "Aspirin/MIGRANES" neural network simulator made available free from the MITRE Corporation.

[9] The sepal length, sepal width, petal length, and petal width were measured on 50 iris specimens from each of three species, Iris setosa, Iris versicolor, and Iris virginica.

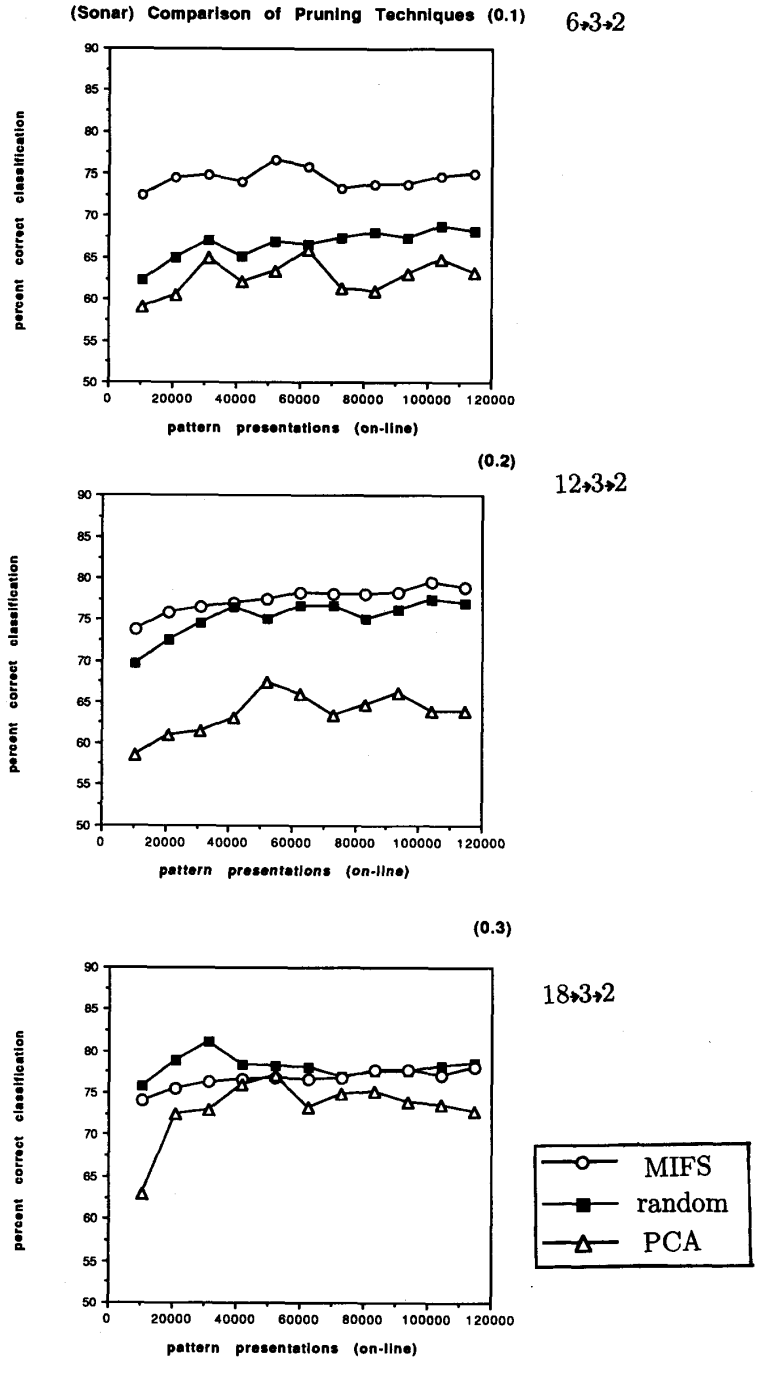


Fig. 7. Generalization curves for the sonar problem, for three values of the dimension of the reduced input vector (6, 12, 18). In each case the selection methods MIFS, PCA and random are compared.

The difference in the amount of the *mutual information* between the set of features and the class, for two different cases (the best case "1010" and the worst case "1100") can be examined by considering Fig. 9. While in the first case the regions corresponding to the different classes are clear (apart from a limited contact zone) in the second case two of the three classes are almost overlapped.

E. Optical Character Recognition

The features for this problem are derived from a real-world task of handwritten digits recognition. The original images are normalized to fit a window of 16 (horizontally) $\times$ 28 (vertically) pixels. The area is then divided into 4 $\times$ 7 nonoverlapping windows of size 4 $\times$ 4, and from each window

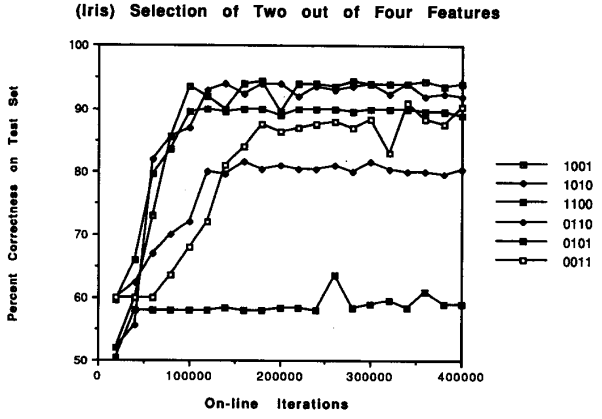


Fig. 8. Generalization curves for the Iris problem. All possible selections of two features are compared.

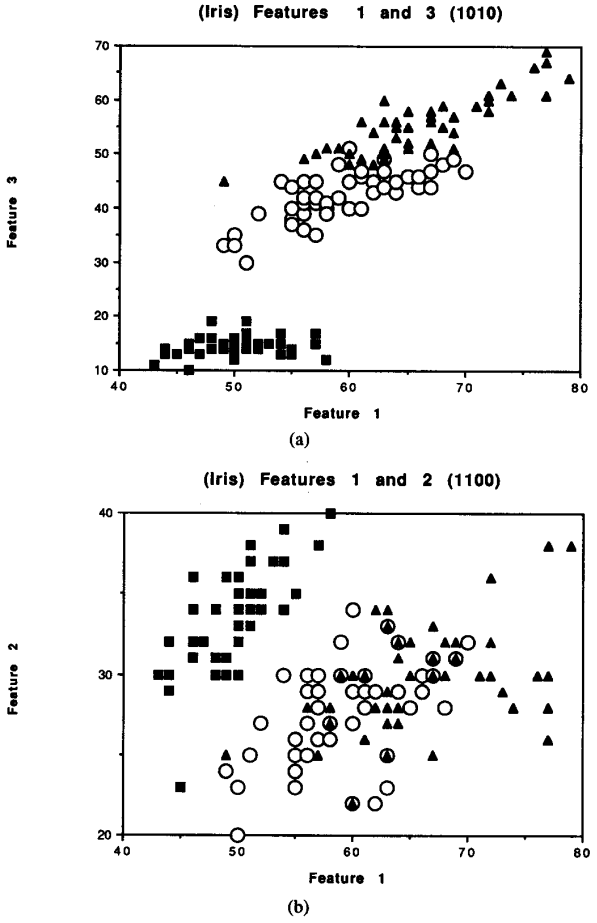


Fig. 9. The Iris classification task. Projection of the original data for the three classes on dimensions 1 and 3 (above) or 1 and 2 (below).

a feature is extracted as the percent of black pixels in the given window. A set of 6496 patterns (equally distributed in the ten classes) is used for the training, a distinct set of 12 981 is used for testing the generalization performance. Here we are not interested in reaching the best accuracy (that demands

TABLE IV
COMPARISON OF FEATURE SELECTION TECHNIQUES FOR THE OPTICAL CHARACTER RECOGNITION PROBLEM WITH BACKPROPAGATION TRAINING. THE DATA FOR THE RANDOM CUT ARE THE AVERAGE AND STANDARD DEVIATION OF 10 TESTS WITH DIFFERENT SEEDS FOR THE RANDOM NUMBER GENERATOR

number of features	MIFS β=0.0	MIFS β=0.5	PCA	random ave. (st.dev.)	prob. $\geq$ MIFS
3	39.4	39.6	40.6	38.03 (3.21)	0.312
6	55.9	66.0	56.2	59.78 (3.96)	0.058
8	61.4	73.9	64.9	68.43 (4.13)	0.092
11	74.0	83.2	79.0	78.33 (3.95)	0.108
14	82.5	88.3	87.4	83.19 (2.05)	0.006
20	90.0	91.9	91.7	90.73 (0.70)	0.047

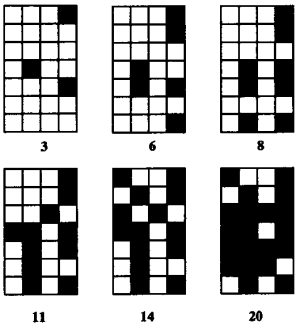


Fig. 10. Feature selection by the MIFS algorithm for the OCR problem. The different figures correspond to increasing numbers of features extracted from 28 windows on the image plane. Latest added features are gray, previously added are black.

the use of rejection schemes, where the uncertain patterns are discarded) but in comparing different feature selection techniques when the classification of *all* patterns is required. The generalization performance of networks trained with on-line backpropagation (learning rate=1.2, momentum=0.0) is 94.7%.

The maximum generalization in this parameter setting is obtained for a number of pattern presentations equal to about 150 000. For a number of presentations larger than 250 000 there is a slight performance reduction caused by over-learning. Patterns are presented to a network with a single hidden layer of 28 units after extracting them randomly from the training set.

In the following tests we execute a total of 250 000 iterations, check the generalization every 50 000 and list the maximum obtained. In Table IV we show the comparison of the MIFS technique with respect to the Principal Component Analysis and to a set of random selections. The position on the image plane of the windows corresponding to the selected features is shown in Fig. 10. Note that the "better" features are preferably located in the top, central, and bottom part (see the cases with 3, 6, and 8 features). This corresponds approximately to the position of the most informative strokes in the image.

In the last column we show the probability that an accuracy greater than or equal to that of MIFS is obtained by using a random cut (in the assumption of a normal distribution of values). It is apparent that tens or hundreds of random cuts have to be tested (by training the network) before reaching

TABLE V
COMPARISON OF FEATURE SELECTION TECHNIQUES FOR THE OPTICAL CHARACTER RECOGNITION PROBLEM WITH LEARNING VECTOR QUANTIZATION TRAINING

number of features	MIFS (β=0.5) + olvq1	random + olvq1
3	32.1	32.29 (4.23)
6	62.9	55.66 (3.17)
8	70.9	61.97 (4.32)
11	81.3	75.53 (3.55)
14	86.7	80.62 (1.46)
20	91.3	90.02 (0.69)

a comparable result. Because the feature selection time of MIFS is negligible with respect to the training time, this amounts to a sizable reduction in CPU resources to obtain a given performance. For the case of 14 features, in the normal distribution hypothesis, approximately 115 random selections (out of a total of $\binom{28}{14} = 40\,116\,600$)) have to be tested to find a performance equivalent to or better than that of MIFS, with a probability greater than 0.5.

The PCA method performs less than the average random cut in some cases (with 6 or 8 features), while it is close to the MIFS results for 14 and 20 features. Let us note that the presence of a β value larger than zero is crucial in order to obtain good results. If the mutual dependencies between features are not taken into account, selecting the features with the highest MI with respect to the output tends to produce a set of redundant features that leaves out useful "complementary" information.

To test the robustness of the MIFS algorithm with respect to different neural net models, we repeated the training and generalization tests with the same feature vectors used for the previous results but using the Learning Vector Quantization technique for training the classifier. The LVQ method is described in [11]. In particular we used an optimized version of the method (OLVQ1), that is part of a software package obtained from the Helsinky University of Technology[10]. In this optimized version, an individual learning rate is assigned to each *codebook vector*[11] and properly adjusted during training.

A performance comparable (although inferior) to that of MLP was obtained with a total of 2000 codebook vectors. These vectors were appropriately initialized and balanced (see the package manual), before executing a total of 20 000 iterations, an empirical number corresponding to the maximum generalization.

While the absolute results for this problem are better when using the MLP neural net (and a large number of codebook vectors has to be used to obtain a near performance), the MIFS technique remains superior in relative terms.

V. CONCLUDING REMARKS

The main motivation for this research was to investigate the practical applicability of the *mutual information* concept from Information Theory for the supervised training of neural networks. In the machine learning literature the *entropy* and *mutual information* concepts are used for example in [17] and [15] to introduce relevant features for learning Boolean formulas with a tree representation. In this case and, in general, in complex recognition tasks one encounters many forms of the "curse of dimensionality" problem (see e.g., [3]): approaches that are suitable for a low pattern dimensionality may become unworkable for large dimension because of unrealistic needs of computation and data. Therefore it is crucial to reduce the input dimensionality of a classification problem either by eliminating features with low information content or high redundance with respect to other features or by constructing more powerful features in the preprocessing phase.

Our objective was less ambitious, because only the first of the above options was considered (leaving the second for the capabilities of the neural net to build complex features from simple ones). We assumed that a set of candidate features with globally sufficient information is available and that the problem is that of extracting from this set a suitable subset that is sufficient for the task, thereby reducing the processing times in the operational phase and, possibly, the training times and the cardinality of the example set needed for a good generalization.

In particular we were interested in the applicability of the *mutual information* measure. For this reason we considered the estimation of the MI from a finite set of samples, showing that the MI for different features is over-estimated in approximately the same way. This estimation is the building block of the MIFS algorithm, where the features are selected in a "greedy" manner, ranking them according to their MI with respect to the class discounted by a term that takes the mutual dependencies into account.

In the neural networks literature, concepts from Information Theory have been used both to construct learning algorithms and to analyze the functionality of the classifier (some examples from the literature have been cited in Section II). The present approach is different from pruning methods acting during the learning phase (e.g., [16], [21]) because the dimensionality reduction is executed before learning starts. The main advantage is that irrelevant features are eliminated from the beginning and that a fast informative feedback about the relevance and dependencies of the different features is available to the developer of a neural net classifier. In addition, it is different from methods that use some form of *entropy* estimation during the learning phase (e.g., [1]), in that the usual backpropagation algorithm is used for learning. Although the availability of sufficient information does not guarantee the convergence of a neural net training algorithm to a satisfactory performance level, we presented some examples in different classification areas where the method is satisfactory.

ACKNOWLEDGMENT

The author wishes to thank Dr. A M. Colla for many fruitful discussions and the referees for their useful comments. The training and testing sets for some experiments were kindly made available by Prof. Scott E. Fahlman at Carnegie Mellon University and Dr. Russell Leighton at the MITRE Signal

[10]The LVQ_PAK Package Version 2.0 was prepared by the LVQ Programming Team of the Helsinki University of Technology and kindly made available through the network at the Internet site cochlea.hut.fi (130.233.168.48), in the directory /pub/lvq_pak.

[11]The *codebook vectors* are the free parameter vectors to be distributed in the input space.

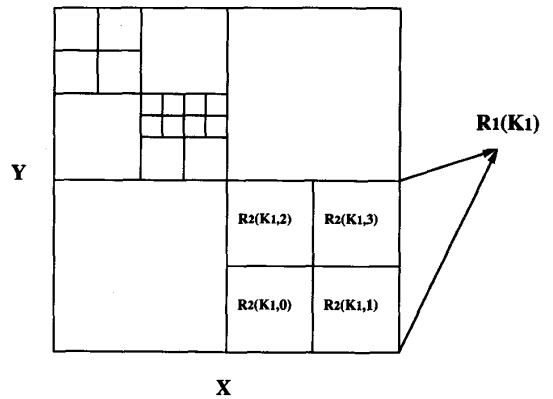


Fig. 11. Recursive partitioning of the X-Y plane executed by Fraser's algorithm. If substructure is found, an element is subdivided into four subelements.

Processing Center. The Learning Vector Quantization program package was developed by the LVQ Programming Team of the Helsinky University of Technology.

APPENDIX
FRASER'S ALGORITHM

Let us start from the definition of the mutual information $I(X,Y)$ between two variables X and Y:

$$I(X,Y) = \int P_{xy}(x,y) \log_2 \left(\frac{P_{xy}(x,y)}{P_x(x)P_y(y)} \right) dx\, dy \quad (19)$$

If the various probability distributions are not known, they can be approximated by a piece-wise constant function by counting the number of events in rectangular boxes. For example, if a box in the $X - Y$ plane of size $\Delta x\ \Delta y$ contains N_{xy} events, the probability density in the region can be estimated by $P_{xy}(x,y) \approx N_{xy}/N_0 \Delta x \Delta y$, where N_0 is the total number of events. The box size at a given point must be large enough to contain a number of points that is sufficient for a robust estimation, but not too large, otherwise part of the structure in the mutual probability density function $P_{xy}(x,y)$ will be cancelled and the mutual information will be underestimated. In general, no single size is appropriate over the whole $X - Y$ plane. Fraser's algorithm is based on an adaptive partition of the plane in which the size of each box is chosen according to the local situation.

Although the algorithm can be modified for a general case, for illustrative purposes it is easier to consider the case where the number of points N_0 is a power of 2, say $N_0 = 2^n$. Let us consider a sequence of partitions of the $X - Y$ plane, such that each partition consists of 4^m boxes $R_m(K_m)$, obtained by dividing each axis into 2^m equi-probable segments. K_m is an index that uniquely identifies one of the 4^m element. It is useful to organize the partition as a tree, so that when an element $R_m(K_m)$ is divided into four parts it generates four children $R_{m+1}(K_m,0); R_{m+1}(K_m,1); R_{m+1}(K_m,2); R_{m+1}(K_m,3)$, see the illustration in Fig. 11.

The approximation of the mutual information corresponding to the mth recursive step, with a partition consisting of 4^m elements, is:

$$I_m = \sum_{K_m} P_{xy}(R_m(K_m)) \log_2 \left[\frac{P_{xy}(R_m(K_m))}{P_x(R_m(K_m))P_y(R_m(K_m))} \right] \quad (20)$$

Let us introduce N_{Km}, equal to the number of events in the $R_m(K_m)$ element of the partition and N_{Kmj} (for $j = 0,1,2,3$) equal to the number of events in the four sectors of elment $R_m(K_m)$ when this is subdivided by cutting both its x and y intervals into two equi-probable parts. After substituting the probability densities with their piece-wise constant approximations and remembering that, because of the subdivision procedure, $P_x(R_m(K_m)) = P_y(R_m(K_m)) = 4^{-m}$, the above expression becomes:

$$\begin{aligned} I_m &= \sum_{K_m} \frac{N_{Km}}{N_0} \left(\log_2 \left(\frac{N_{Km}}{N_0} \right) + m \log_2(4) \right) \quad (21) \\ &= \frac{1}{N_0} \sum_{K_m} (N_{Km} \log_2(N_{Km}) + N_{Km}\ m\ \log_2(4)) \\ &\quad - \log_2(N_0) \end{aligned}$$

When a single element $R_m(K_m)$ of the partition is subdivided into four sectors, its contribution to the mutual information changes from:

$$(N_{Km} \log_2(N_{Km}) + N_{Km}\ m\ \log_2(4))$$

to:

$$\begin{aligned} &\sum_{j=0}^{3} (N_{Kmj} \log_2(N_{Kmj}) + N_{Kmj}\ (m+1)\ \log_2(4)) \\ &= \sum_{j=0}^{3} (N_{Kmj} \log_2(N_{Kmj})) + N_{Km}(m+1) \log_2(4) \end{aligned}$$

where the fact that $\sum_{j=0}^{3} N_{Kmj} = N_{Km}$ has been used.

Starting from (21), and using the above result, it is immediate to check that the mutual information can be estimated by the following formula, that uses the recursive function $F()$ introduced in [6]:

$$I(X,Y) = \frac{F(R_0(K_0))}{N_0} - \log_2(N_0) \quad (22)$$

where the function $F()$ takes a partition element as argument and returns a real value (a floating point number). If the element has no substructure:

$$F(R_m(K_m)) = N_{Km} \log_2(N_{Km})$$

where N_{Km} is the number of events contained in the element, otherwise the function calls itself four times in a recursive way, and returns:

$$F(R_m(K_m)) = N_{Km} \log_2(4) + \sum_{j=0}^{3} F(R_{m+1}(K_m,j))$$

The χ-square test is used to check for substructure. Let us introduce the following variables, that count the number of

events in the initial element and in the elements of the first and second subdivision:

$$N \equiv N(R_m(K_m))$$

$$a_i \equiv N(R_{m+1}(K_m, i))$$

$$b_{ij} \equiv N(R_{m+2}(K_m, i, j))$$

The null hypothesis that $P_{xy}(x, y)$ is flat over $R_m(K_m)$ is disproved if at least one of the following inequalities fails (reduced χ-square statistics and 20% confidence levels):

$$\chi_3^2 \equiv \left[\frac{16}{9N}\sum_{i=0}^{3}(a_i - \frac{N}{4})^2\right] < 1.547$$

$$\chi_{15}^2 \equiv \left[\frac{256}{225N}\sum_{i,j=0}^{3}(b_{ij} - \frac{N}{16})^2\right] < 1.287$$

To simplify the counting operations needed by the algorithm a change of variables is executed that maps the arrays of events x_i and y_j into the integers in the $[0, 2^n - 1]$ interval ($n = \log_2 N_0$). The arrays are sorted into ascending numerical order and a value x_i is mapped into its position in the sorted array. The same procedure is applied to y_i.

In our implementation the sorting is executed by the "heapsort" algorithm (see for example [22]) because its computational complexity is guaranteed to be of order $N_0 \log N_0$ not only in the average but also in the worst case. Besides, no additional storage is required (the sorting is done "in place"). The "quicksort" algorithm used in [6] has a worst-case complexity of N_0^2 operations, that may be excessive for some computations, although in the average it requires order $N_0 \log N_0$ operations (note that the average case may not be that encountered in practical cases).

Because of the tree structure, at most order $N_0 \log N_0$ recursive calls are executed and therefore the total running time is guaranteed to be of order $N_0 \log N_0$ operations. The slow growth with respect to N_0 make this algorithm an efficient one even for very large number of events. On a current Unix workstation the actual computing time is about one second for N_0 equal to 8192.

REFERENCES

[1] M. Bichsel and P. Seitz, "Minimum class entropy: A maximum information approach to layered networks," *Neural Networks*, 2:133–141, 1989.

[2] J. S. Bridle, "Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters," in, *Advances in Neural Information Processing Systems*, vol. 2, D. S. Touretzky, ed. San Mateo, CA: Morgan Kaufmann, 1990, pp. 211–217.

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

[4] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of Eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

[5] A. M. Fraser, "Reconstructiong attractors form scalar time series: a comparison of singular system and redundancy criteria," *Physica D*, vol. 34, pp. 391–404, 1989.

[6] A. M. Fraser and H. L. Swinney, "Independent coordinates for strange attractors from mutual information," *Physical Review A*, vol. 33, no. 2, pp. 1134–1140, 1986.

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

[8] R. P. Gorman and T. J. Sejnowski, "Analysis of hidden units in a layered network trained to classify sonar targets," *Neural Networks*, vol. 1, pp. 75–89, 1988.

[9] F. Kanaya and K. Nakagawa, "On the practical implication of mutual information for statistical decisionmaking," *IEEE Trans. Information Theory*, vol. 37, pp. 1151–1156, 1991.

[10] E. D. Karnin, "A simple procedure for pruning back-propagation trained neural networks," *IEEE Trans. Neural Networks*, vol. 1, pp. 239–242, 1990.

[11] T. Kohonen, "The self-organizing map," *Proc. IEEE*, vol. 78, pp. 1464–1480, 1990.

[12] W. Li, "Mutual information functions versus correlation functions," *J. Stat. Phys.*, vol. 60, no. 5/6, pp. 823–837, 1990.

[13] R. Linsker, "How to generate ordered maps by maximizing the mutual information between input and output signals," *Neural Computation*, vol. 1, no. 3, pp. 402–411, 1989.

[14] H. A. Malki and A. Moghaddamjoo, "Using the Karhunen–Loe've transformation in the back-propagation training algorithm," *IEEE Trans. Neural Networks*, vol. 2, pp. 162–165, 1990.

[15] A. Mar.etti-Spaccamela and M. Protasi, "Learning Boolean formulae using decision trees," in *Proc. 3rd Workshop on Parallel Architectures and Neural Networks*, Vietri, Italy, 1990, pp. 797–802.

[16] M. C. Mozer and P. Smolensky, "Skeletonization: A technique for trimming the fat from a neural network via relevance assessment," in *Advances in Neural Information Processing Systems*, vol. 1, D. S. Touretzky, ed. San Mateo, CA: Morgan Kaufmann, 1989, pp. 107–115.

[17] G. Pagallo and D. Haussler, "Boolean feature discovery in empirical learning," *Machine Learning*, vol. 5, pp. 71–99, 1990.

[18] E. Pernot and F. Vallet, "Determining the relevant parameters for the classification on a multi-layer perceptron: Application to radar data," in *Proc. 1991 Int. Conf. Artificial Neural Networks ICANN-91*, Espoo, Finland, June 1991, pp. 797–802.

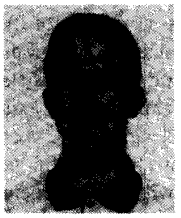
[19] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* , vol. 1, D. E. Rumelhart and J. L. McClelland, ed. Cambridge, MA: MIT Press, Bradford Books, 1987, pp. 318–362.

[20] C. E. Shannon and W. Weaver, *The Mathematical Theory of Communication*. Urbana, IL: University of Illinois Press, 1949.

[21] A. S. Weigend, D. E. Rumelhart, and B. A. Huberman, "Generalization by weight-elimination with application to forecasting," in *Advances in Neural Information Processing Systems*, vol. 3, R. P. Lippmann, J. Moody, and D. S. Touretzky, ed. San Mateo, CA: Morgan Kaufmann, 1991, pp. 875–882.

[22] S. A. Teukolsky, W. H. Press, B. P. Flannery, and W. T. Wetterling, *Numerical Recipes in C*. Cambridge, MA: Cambridge University Press, 1988.

[23] R. S. Zemel and G. E. Hinton, "Discovering view-point invariant relationships that characterize objects," in *Advances in Neural Information Processing Systems*, vol. 3, R. P. Lippmann, J. Moody, and D. S. Touretzky, ed. San Mateo, CA: Morgan Kaufmann, 1991, pp. 299–305.



Roberto Battitii received the B.S. and M.S. degrees in physics from the University of Trento, Italy in 1985 and the Ph.D. degree in computation and neural systems from the California Institute of Technology, Pasadena, CA in 1990.

He then became a consultant for the industrial application of neural networks and concurrent processing, a faculty member at the University of Trento, (June 1991), and an Associate of Istituto Nazionale di Fisica Nucleare. His current interests are the study of efficient learning techniques and algorithms for global optimization that can be implemented with massively parallel architectures and the use of neural nets for pattern recognition. Dr. Battiti is a member of the IEEE Computer Society.