Pattern Recognition 41 (2008) 3706 – 3719

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Feature selection using localized generalization error for supervised classification problems using RBFNN

Wing W.Y. Ng[a,b,*], Daniel S. Yeung[a,b], Michael Firth[c], Eric C.C. Tsang[b], Xi-Zhao Wang[d]

[a]School of Computer Science and Engineering, South China University of Technology, Guangzhou 510640, China
[b]Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
[c]Department of Finance and Insurance, Lingnan University, Hong Kong
[d]Machine Learning Center, Faculty of Mathematics and Computer Science, Hebei University, Baoding 071002, China

A R T I C L E   I N F O

Article history:
Received 1 March 2007
Received in revised form 1 March 2008
Accepted 5 May 2008

Keywords:
Feature selection
Neural network
Generalization error
RBFNN

A B S T R A C T

A pattern classification problem usually involves using high-dimensional features that make the classifier very complex and difficult to train. With no feature reduction, both training accuracy and generalization capability will suffer. This paper proposes a novel hybrid filter–wrapper-type feature subset selection methodology using a localized generalization error model. The localized generalization error model for a radial basis function neural network bounds from above the generalization error for unseen samples located within a neighborhood of the training samples. Iteratively, the feature making the smallest contribution to the generalization error bound is removed. Moreover, the novel feature selection method is independent of the sample size and is computationally fast. The experimental results show that the proposed method consistently removes large percentages of features with statistically insignificant loss of testing accuracy for unseen samples. In the experiments for two of the datasets, the classifiers built using feature subsets with 90% of features removed by our proposed approach yield average testing accuracies higher than those trained using the full set of features. Finally, we corroborate the efficacy of the model by using it to predict corporate bankruptcies in the US.

1. Introduction

With the availability of fast computers, broadband Internet, and cheap, high capacity storage, datasets have become ever larger. Usually, domain knowledge and personal bias influence the choice of features. Although these parameters may not fully describe the problem, some parameters may be included just for fear of losing something useful. When the number of parameters (input features) of the dataset becomes large, the pattern classification systems trained for differentiating the sample points into different classes also get more complex. On the other hand, if it is not necessary to collect so many input features, the cost of data collection and storage will be reduced.

A major problem in pattern classification is how to build a simple classifier that has good performance. By "good performance" we mean a system that can be quickly trained, is highly accurate and responds quickly to future unseen samples, and is easily understood by people. Perhaps the most straightforward way to reduce the complexity of a classifier is to reduce the number of input features.

Given the training dataset $D = \{(\mathbf{x}_b, F(\mathbf{x}_b))\}_{b=1}^{N}$ consisting of N training samples $(\mathbf{x}_b)$ with F denoting the unknown input–output mapping of the classification problem that one would like to approximate using a classifier (e.g. a neural network), the training error (R_{emp}) and generalization error (R_{true}) for the entire input space (T) of the classifier f_θ are defined as

$$R_{emp} = \frac{1}{N}\sum_{b=1}^{N}(F(\mathbf{x}_b) - f_\theta(\mathbf{x}_b))^2 \tag{1}$$

$$R_{true} = \int_T (F(\mathbf{x}) - f_\theta(\mathbf{x}))^2 p(\mathbf{x})\,d\mathbf{x} \tag{2}$$

where $p(\mathbf{x})$ denotes the true unknown probability density function of $\mathbf{x}$, and θ denotes the set of parameters in the classifier f_θ. The ultimate goal of training a classifier is to minimize the generalization error for unseen samples (i.e. minimizing the differences between the real unknown input–output mapping function and the mapping approximated by f_θ). Moreover the ultimate goal of feature selection is to maintain the classifier's generalization capability using a reduced

* Corresponding author at: School of Computer Science and Engineering, South China University of Technology, Guangzhou 510640, China.
E-mail addresses: wingng@ieee.org (W.W.Y. Ng), csdaniel@comp.polyu.edu.hk (D.S. Yeung), mafirth@ln.edu.hk (M. Firth), csetsang@comp.polyu.edu.hk (E.C.C. Tsang), wangxz@mail.hbu.edu.cn (X.-Z. Wang).

0031-3203/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2008.05.004

set of features. Classifiers (e.g. neural networks) are usually not expected to recognize unseen samples that are too different from the training samples. Therefore, assessing the generalization capability of a classifier for those unseen samples may be counter-productive to classifier learning. So, Ng et al. [1,2] proposed a localized generalization error model for bounding the generalization error (R^*_{SM}) for a classifier for unseen samples similar to the training samples. In our proposed feature selection method (R_{SM}FS), we remove the feature subset that yields the smallest contribution to the R^*_{SM}. In terms of probability, the classifier trained using the reduced feature subset will not lose its generalization capability if R^*_{SM} remains unchanged. In this paper, the widely adopted radial basis function neural networks (RBFNNs) with Gaussian basis function [3,4] will be used to demonstrate the R_{SM}FS method.

A brief literature review is presented in Section 2. In Section 3 we describe the localized generalization error model. The novel feature selection method R_{SM}FS is presented in Section 4, while experimental results are shown in Section 5. Section 6 concludes the paper.

2. Existing feature selection methods

Broadly speaking, the number of input features is reduced using three feature selection approaches: filters, wrappers, and embedded [5–7]. Under certain circumstances in the learning process of a decision tree, some features are ignored in the final decision tree if they have a minor influence on the classification [8]. This is a special case of feature selection and we will not discuss it in this work. In the following two sub-sections, we will introduce the filter and wrapper approaches.

In Fig. 1, we illustrate the relationship between different relevant measures for feature selection. A relevant measure is employed in each feature selection method and we will have more discussion on each of these measures in Sections 2.1 and 2.2.

Principal component analysis [9,10] and other transformation-based feature reduction methods are not discussed in this paper because they do not select the features from the original feature set. These methods transform the feature set into a lower-dimensional feature vector by combining several features. Transformation-based feature reduction methods do not reduce the cost of future sample collection and storage. Moreover, the newly created feature vector is usually difficult to interpret. For example, in the physiology field, a feature vector may be composed of blood pressure times the square of body height and this kind of feature does not help people understand the problem.

2.1. Filter approaches

Filter approaches make use of statistical information of the dataset to carry out feature selection and are independent of the

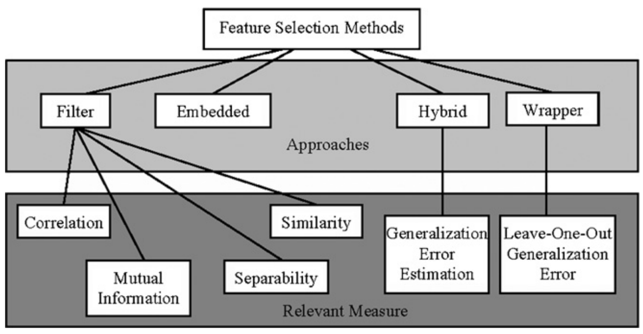

Fig. 1. Relationship between relevant measures for feature selection.

classification system. These approaches rely on the definition of a relevant measure.

The simplest measure may be the correlation between the input and output using the correlation coefficient [6,7]. The absolute value of the correlation coefficient may be used because we may want to focus on the magnitude of the correlation between the input feature and the output. The major drawback is that it ignores any nonlinear correlation between input and output.

This problem could be solved by using the mutual information to replace the correlation coefficient. In mutual information approaches [11,12], the mutual information between the inputs and outputs are computed and sorted. The input feature that yields the maximum mutual information to the outputs is selected. Then the mutual information between the outputs and also between the selected feature subsets is computed. The feature yielding the maximum mutual information is added into the selected subset. These procedures are repeated until a specified number of features are reached. This approach has a sound theoretical underpinning, yet the computation of the probability density function between features and outputs is expensive. Kwak et al. [13] improved the approach by using the Parzen Window to estimate the density functions, but the computation effort is still high for a dataset with large numbers of features and samples In this work, we adopt the definition of mutual information proposed in Ref. [11].

Mitra [14] proposed using a similarity measure between input features. In his work, features are grouped by similarity and only one feature in each group is selected. This method does not take into account the performance of the features and simply deletes similar features. As a result, the performance of the feature selection is determined by choosing the number of groups and the similarity measure.

The authors in Refs. [15,16] applied the class separability measure to select feature subsets. The feature with the maximum class separability is selected first. The next feature with the maximum class separability will then be selected after removing the first one. The process stops when no more features provide class separability larger than a given threshold. This approach considers the training classification accuracy indirectly. However it cannot deal with the case in which the samples from one class are surrounded by samples from another class. In Ref. [17], the authors proposed removing features that are inconsistent to the class label (i.e. could not separate samples from two classes). However, the point-wise comparison makes the method infeasible for a large dataset.

One may observe that the above filtering approaches require users to determine the number of features selected, rather than providing stopping criteria. Furthermore, the generalization performance of the classifier is not a consideration of the filtering feature selection methodologies, even though it is the ultimate goal of building classifiers. Yet, they are free from the bias of classifier training. The feature selection criteria presented in this section relate to the training accuracy indirectly.

2.2. Wrapper approaches

Wrapper feature selection methodologies combine both the feature selection and output of the classification system into a single system [18]. Most of the wrappers employ the Leave-One-Out searching method [19] which, in each step, evaluates the training accuracy when one of the features is left out, and then removes the feature yielding the least reduction in accuracy. The Leave-One-Out wrapper feature selection methodology can be applied to any classification system.

However, the procedures mentioned above only make use of the training accuracy as the relevant measure. Since a classifier is built, the validation accuracy is used to ensure that the classification

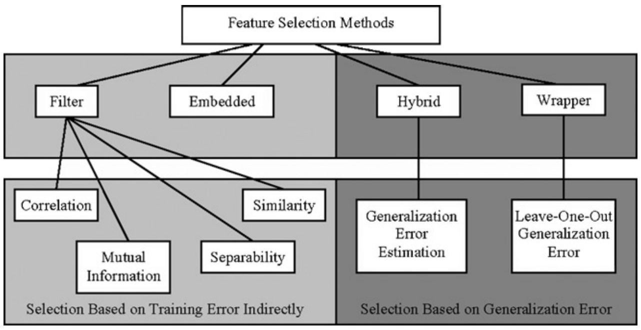
Feature Selection Methods
Filter
Embedded
Hybrid
Wrapper
Correlation
Similarity
Mutual Information
Separability
Generalization Error Estimation
Leave-One-Out Generalization Error
Selection Based on Training Error Indirectly
Selection Based on Generalization Error

Fig. 2. Relationship between accuracy and relevant measures.

accuracy of unseen samples is retained. One may find the generalization error using k-fold cross-validation (CV). In Ref. [20], five folds are suggested. For a k-fold CV, the training dataset is divided into k disjoint partitions and the pth partition is reserved as the validation set which will not participate in the classifier training. Then, k classifiers are trained without the ith feature for the evaluation of the relevance of the ith feature and the average of the validation error using the left out partition is used as the relevant measure. This CV relevant measure is used to estimate the generalization error of the classifier without the use of the ith feature. For example, if the average CV error for the classifiers trained with the ith feature omitted yields the least loss, the ith feature will be removed. In the next round, these steps will be repeated with one less feature. The steps are repeated until no more features can be removed because none of the classifiers gives good results after the removal of any feature or if the pre-selected number of features is reached. Thus, nk classifiers are required for a problem with n features. Certainly, if k is small, the estimation of the generalization error may be poor because a larger portion of training samples are reserved and this makes a significant difference from the original training dataset. In contrast, if k is large, the number of classifier trainings and computational effort will be huge.

There are two major drawbacks of wrapper approaches: they are computationally expensive for datasets consisting of large numbers of features, and they are not suitable for datasets with small numbers of samples Wrapper approaches are usually brute force methods. The methodologies mentioned above require extremely computationally intensive nonlinear regression or optimization of high-dimensional nonlinear systems. They are not scalable when the numbers of input features and training samples are large. Moreover, the use of a validation set reduces the number of training samples for classifier training.

2.3. Motivation of the R_{SM}FS

In general, the major difference between filter and wrapper approaches is that the relevant measures in filter approaches consider only the training error indirectly, while those in wrapper approaches consider the generalization error directly (Fig. 2). The ultimate goal of feature selection is to maintain the generalization capability of the classifier using a reduced subset of features. Therefore, wrapper approaches should be a better choice. However, they are very time consuming and computationally infeasible for a dataset consisting of large numbers of features and samples. These issues motivate us to propose the R_{SM}FS to estimate the generalization error for removing a feature instead of finding its real generalization error by training a set of new classifiers. Thus, only one trained classifier is required for selecting one feature. Moreover, we estimate the generalization error based on the training dataset using the R^*_{SM} instead of the validation error using a reserved portion of a training dataset.

3. Localized generalization error model

In this work, we concentrate our discussion on the use of RBFNN as a classifier (f_θ), which is trained by the minimization of mean-square error (MSE) that indicates how good the RBFNN is when approximating the true unknown input–output mapping function (F). The localized generalization error bound (R^*_{SM}) is an upper bound of the MSE of those unseen samples that have features similar to the training samples (i.e., having a distance smaller than a constant Q in the input space) [1,2]. We are not interested in the error for the unseen samples that are very dissimilar to the training samples because the generalization performance of the classifier on those samples is usually poor. We do not have any information about unseen samples that are very different from the training dataset and the classifier cannot learn in this part of the input space. The error of extrapolation in such situations is expected to be high and the predictions will be misleading Therefore, this may be counterproductive to assess the generalization performance of the classifier on them and minimize the generalization error on such samples.

The Q-neighborhood ($S_Q(\mathbf{x}_b)$) of a training sample $\mathbf{x}_b$ is defined as $S_Q(\mathbf{x}_b)=\{\mathbf{x}=\mathbf{x}_b+\Delta\mathbf{x}\}$ for all $\Delta\mathbf{x}$ that fulfils $0<|\Delta x_i|\leqslant Q\ \forall i=1,\dots,n$, where n denotes the number of features, Q is a real value and $\Delta\mathbf{x}=(\Delta x_1,\dots,\Delta x_n)'$. The Q-union of the whole training dataset (S_Q) is defined to be the union of all $S_Q(\mathbf{x}_b)$. We define R_{SM} to be the generalization error for the unseen samples located within the Q-union (i.e., the shaded region in Fig. 3). With probability $1-\eta$, we have [1,2]:

$$\begin{aligned} R_{SM}(Q) &= \int_{S_Q} (f_\theta(\mathbf{x})-F(\mathbf{x}))^2 p(\mathbf{x})\,\mathrm{d}\mathbf{x} \\ &\leqslant \left(\sqrt{E_{S_Q}((\Delta y)^2)}+\sqrt{R_{emp}}+A\right)^2+\varepsilon \\ &= R^*_{SM}(Q) \end{aligned} \tag{3}$$

where $\Delta y = f_\theta(\mathbf{x})-f_\theta(\mathbf{x}_b)$, $\varepsilon = B\sqrt{\ln\eta/(-2N)}$, $E_{S_Q}((\Delta y)^2)$, η, A and B denote the stochastic sensitivity measure (ST-SM), the confidence of the bound, the difference between the maximum and minimum values of the target output (F), and the maximum value of the MSE, respectively. A and B can be fixed when the dataset is given.

The R^*_{SM} in Eq. (3) can be applied to any type of classifier whose ST-SM is defined. In particular, with probability $1-\eta$, we have the R^*_{SM} for RBFNN [1,2] as follows:

$$R^*_{SM}(Q) \approx \left(\sqrt{\frac{1}{3}Q^2\sum_{j=1}^{M}\upsilon_j+\frac{0.2}{9}Q^4 n\sum_{j=1}^{M}\zeta_j}+\sqrt{R_{emp}}+A\right)^2+\varepsilon \tag{4}$$

where $\mathbf{x}=(x_1,x_2,\dots,x_n)'$, $\varphi_j=(w_j)^2\exp((Var(s_j)/2v_j^4)-(E(s_j)/v_j^2))$, $E(s_j)=\sum_{i=1}^{n}(\sigma_{x_i}^2+(\mu_{x_i}-u_{ji})^2)$, $Var(s_j)=\sum_{i=1}^{n}(E_D[(x_i-\mu_{x_i})^4]-(\sigma_{x_i}^2)^2+4E_D[(x_i-\mu_{x_i})^3](\mu_{x_i}-u_{ji})+4\sigma_{x_i}^2(\mu_{x_i}-u_{ji})^2)$, $s_j=\|\mathbf{x}-\mathbf{u}_j\|^2$, $\upsilon_j=\varphi_j(\sum_{i=1}^{n}(\sigma_{x_i}^2+(\mu_{x_i}-u_{ji})^2)/v_j^4)$, $\zeta_j=\varphi_j/v_j^4$, M denotes the number of hidden neurons in the RBFNN, w_j, v_j and $\mathbf{u}_j=(u_{j1},u_{j2},\dots,u_{jn})'$ denote the connection weight, width and center position of the jth hidden neuron of the RBFNN, and μ_{x_i} and $\sigma_{x_i}^2$ denote the mean and variance of the ith input feature, respectively.

For every trained RBFNN, one could compute the maximum Q-value in which the R^*_{SM} bound is less than or equal to a threshold α. For example, if $F(\mathbf{x})\in\{(o_1,o_2,\dots,o_K)\}$ for a K-class problem, then $o_k=1$ and $\|F(\mathbf{x})\|=1$ for a sample in class k. The threshold α could be selected to be 0.25, which is the threshold of the squared error between the classifier output and the target output of a sample classified correctly. So, the Q-value indicates the coverage of the unseen samples whose generalization error in MSE is less than α. Therefore, a larger Q indicates better generalization of a classifier in a prob-

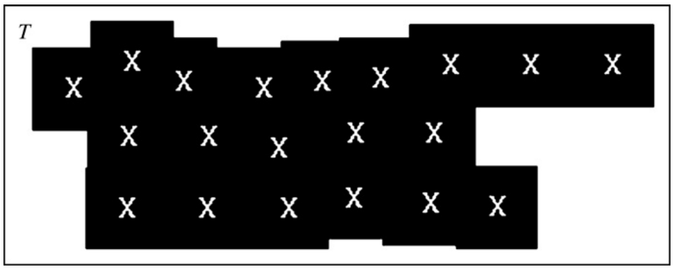

Fig. 3. An illustration of Q-union with 20 training samples. The X marks the elements of the training samples and any other points in the shaded area are the unseen samples.

ability sense [1,2]. Therefore, for two classifiers yielding the same R^*_{SM} with different Q-values, the one that yields a larger Q-value has better generalization capability. The Q-value is computed by solving the following quadratic equation:

$$Q^4 \frac{0.2}{3} N \sum_{j=1}^{M} \zeta_j + Q^2 \sum_{j=1}^{M} \upsilon_j - 3(\sqrt{\alpha - \varepsilon} - \sqrt{R_{emp}} - A)^2 = 0 \tag{5}$$

There is a maximum of four solutions for Eq. (5) and the only one real non-negative solution will be used as the final result.

4. Feature selection using the localized generalization error

We applied the R^*_{SM} to RBFNN architecture selection problems [1] and image classification problems [2]. In this paper, we focus on the use of the R^*_{SM} to select the feature subset for pattern classification problems (i.e. R_{SM}FS), for RBFNN. We define the irrelevant features to be those features yielding the smallest contribution to the R^*_{SM}. By removing these irrelevant features, one could build a classifier with a smaller loss or even no loss in generalization performance with reduced classifier complexity. Let the candidate feature set for feature selection be CFS, which is initially equal to the full set of features ($CFS = \{1, 2, \ldots, n\}$). We formulate this feature selection method as an optimization problem:

$$\arg \min_{Z \subseteq CFS} R^*_{SM}(Q) - R^*_{SM}(Q, Z) \tag{6}$$

where the $R^*_{SM}(Q, Z)$ indicates the R^*_{SM} of holding constant the values of unseen samples of features in Z and are equal to the mean values of those samples in the training dataset (i.e., variances equal to zero). Therefore, the features in Z could be replaced by a constant and thus could be removed. By doing so, $R^*_{SM}(Q) - R^*_{SM}(Q, Z)$ denotes the differences in the generalization error bounds between all the unseen samples located within the neighborhood of training samples with and without the variation in the values of the features in Z. In the case of RBFNN, with probability $1 - \eta$, we have

$$R^*_{SM}(Q, Z) = \left(\sqrt{\frac{1}{3} Q^2 \sum_{j=1}^{M} \upsilon_j(Z) + \frac{0.2}{9} Q^4 (n - l) \sum_{j=1}^{M} \zeta_j} + \sqrt{R_{emp}} + A \right)^2 + \varepsilon \tag{7}$$

where $\upsilon_j(Z) = \varphi_j(\sum_{i=1, i \notin Z}^{n} (\sigma^2_{x_i} + (\mu_{x_i} - u_{ji})^2)/v_j^4)$, Z denotes the set of candidate features removed from the full set of features ($CFS = \{1, 2, \ldots, n\}$) and l denotes the number of features in Z.

For example, if we have 10 features and the features that we are going to test for irrelevance are the 3rd and 5th features, we first compute the $R^*_{SM}(Q)$ for the classifier with unseen samples different from the training samples in all features. Then, we compute the $R^*_{SM}(Q, \{3, 5\})$ for the classifier with unseen samples different from training samples in all features except the 3rd and 5th features. The combined influence of the 3rd and 5th features to the classifier's generalization error is computed using $R^*_{SM}(Q) - R^*_{SM}(Q, \{3, 5\})$.

4.1. Feature subset selection algorithm

Unfortunately, the optimal feature subset found by solving Problem (6) requires comparing all the possible combinations of features in the power set of the full feature set. If there are n features, there will be 2^n possible combinations and this is computationally prohibitive. So, in this paper, we employ a sequential backward search for the feature subset selection (i.e., we start with the classifier trained using a full set of features and remove the most irrelevant features one by one). This is preferred because it requires only n classifiers to be trained and we can compare the feature relevance based on the classifier trained involving the candidate feature and R^*_{SM}. In contrast, sequential forward search still requires $(n^2 + n)/2$ classifiers to be trained. First, one selects the most irrelevant feature by solving

$$\arg \min_{\{z\} \in CFS} R^*_{SM}(Q) - R^*_{SM}(Q, \{z\}) \tag{8}$$

Only n comparisons and one trained classifier are required to solve Problem (8). Then we add the zth feature to the irrelevant feature set (IFS) and remove this feature from the candidate feature set CFS. After that, we iteratively add the most irrelevant feature to the IFS using a new classifier trained with CFS. The next most irrelevant feature is selected by

$$\arg \min_{\{z\} \in CFS} R^*_{SM}(Q) - R^*_{SM}(Q, (\{z\} \cup IFS)) \tag{9}$$

By iteratively solving Problem (9), one obtains a list of irrelevant features in the descending order of its irrelevance and a set of classifiers trained with different numbers of features.

The same Q-value should be used for all RBFNN and feature evaluations, such that all the features and RBFNNs are evaluated using the same set of unseen samples. We may point out that if the Q-value adopted is too small, the unseen samples being considered by the R^*_{SM} may be very few and thus may not be representative. The Q-value could be selected based on domain knowledge. For example, if the input space is in [0,1], a Q-value which is equal to 0.1 represents that the R_{SM}FS considers the unseen samples that deviate from the training samples by no more than 10%. On the other hand, this unique Q-value (with a parameter $\alpha = 0.25$) could be found by Eq. (5) and the RBFNN is trained using the full set of features. In this way, we eliminate the bias of choosing Q-value and this will be adopted in our experiments.

The procedures of implementing the R_{SM}FS are as follows:

1. Initialize the CFS to be equal to the full set of features, and the IFS to be an empty set,
2. Train the classifier using the dataset with features in CFS,
3. Compute the $R^*_{SM}(Q)$ for the classifier trained in Step 2,
4. Compute the $R^*_{SM}(Q) - R^*_{SM}(Q, \{z\})$ for every feature in CFS,
5. Add the zth feature to the IFS and remove it from the CFS if the $R^*_{SM}(Q) - R^*_{SM}(Q, \{z\})$ for the zth feature is the smallest among all choices,
6. If CFS is not empty, go to Step 2. Otherwise end the feature selection process.

In Step 2, we adopt the twostage training method for RBFNN [3,4]. We first perform unsupervised training with k-means for samples in each class to find the centers and widths of the hidden neurons in RBFNN. Then, the centers of all classes are combined to build the RBFNN and we compute the connection weights using the pseudo-inverse method. To avoid the rank deficiency problem in finding the pseudo-inverse of the output matrix of the RBFNN hidden neurons, a diagonal matrix with very small constant values is added to the output matrix [3]. One may notice that the R_{SM}FS does not depend on the training algorithm of the classifier and other training methods may be adopted in Step 2.

4.2. Generalization performance of the classifier trained with selected feature subset

The most important problem with feature selection, perhaps, is the loss in generalization performance of the classifier built using the reduced feature subset. As mentioned before, the Q value for all the computations of R^*_{SM} should be the same value, such that we evaluate all the feature subsets using the same input sub-space (i.e., S_Q). In the R_{SM}FS, we train a classifier with the full set of features and then compare the R^*_{SM} of this classifier with input samples without considering the zth feature. From Eq. (3), the R^*_{SM} is computed based on the training error, ST-SM and two constants (A and ε). The ε is cancelled out in Eqs. (8) and (9); therefore, it does not affect the choice of input feature. Moreover, the training error and constant A are fixed in each iteration of the feature selection; therefore these two parameters do not participate in the selection of features. The ST-SM term (the first term in Eq. (7)) indicates the contribution of the feature set to the classifier outputs by computing the average squared classifier outputs differences between the training samples and unseen samples located within the Q-Union. In $R^*_{SM}(Q,\{z\})$, the zth feature's value is replaced by a constant, and therefore we compute the R^*_{SM} without the contribution of the zth feature. If $R^*_{SM}(Q) - R^*_{SM}(Q,\{z\})$ is zero, the contribution of the zth feature to the classifier outputs is zero. Thus, this feature is insignificant and we could remove it without affecting either the generalization error bound of the classifier or the classification results of the classifier. In practice, one usually finds non-zero differences and the feature yielding the smallest $R^*_{SM}(Q) - R^*_{SM}(Q,\{z\})$ is removed. Therefore, the feature yielding the smallest contribution to the classifier is removed. In contrast, if the zth feature yields a large $R^*_{SM}(Q) - R^*_{SM}(Q,\{z\})$ value, its influence to both the generalization error bound and classifier outputs is big. Therefore, the classifier trained by using the feature subset without this feature may change radically.

Fig. 4 shows the average testing accuracies of the classifier (with 10 independent runs) trained using the full set of features and the reduced feature subsets in iterations for the UCI Wine dataset. We find that the average testing accuracies for the classifier trained with more than four features are not worse than the one trained using all of the features. When there are only three features left, the average testing accuracies drop about 0.5%, which is insignificant. The $R^*_{SM}(Q) - R^*_{SM}(Q,\{z\})$ become very large when there are only two features left and the average testing accuracies drop significantly.

4.3. Stopping criterion for R_{SM}FS

Usually, one stops the feature selection algorithm when a pre-selected number of features are achieved [11]. On the other hand, one may stop the feature selection when the training accuracy of the classifiers trained using the current selected feature subset drops significantly [12,19]. However, by assuming that we do not have knowledge about the future unseen samples, this could not provide a theoretical guarantee on the generalization capability of the classifier. In contrast, in the R_{SM}FS method, if the minimum $R^*_{SM}(Q) - R^*_{SM}(Q,\{z\})$ value is either very large or larger than a pre-selected threshold, it indicates that one should stop the feature selection process because further removal of a feature leads to significant changes in both the generalization error bound and classifier outputs. However, the selection of the stopping threshold requires an ad hoc choice and domain knowledge about the given classification problem. Therefore, we ignore the stopping criterion in this work and remove the feature iteratively to show the change in average testing accuracies. This lends itself to important future research to find a problem-independent stopping threshold.

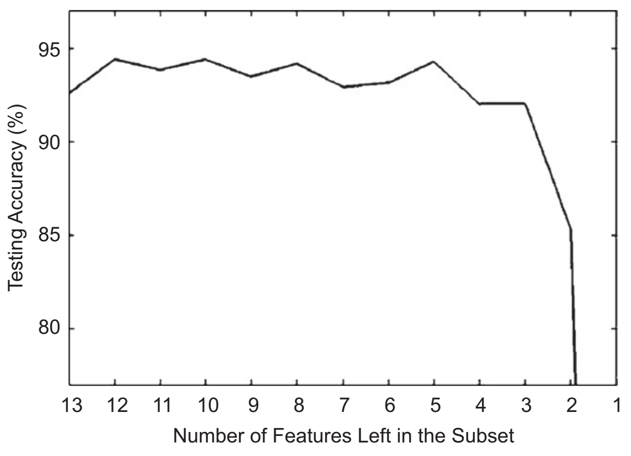

Fig. 4. Average testing accuracies of the classifier trained with reduced feature subsets.

4.4. R_{SM}FS—hybrid filter and wrapper

The R_{SM}FS feature selection method enjoys the benefits of both the wrapper and filter approaches. It uses a trained classifier to start the feature selection. However, it selects the features based on an estimated generalization error bound. This estimation reduces the time-consuming classifier training and provides theoretical justification for the selected feature subset, instead of the widely applied empirical brute force Leave-One-Out feature selection method. Wrapper approaches, including R_{SM}FS method, have classifier bias, yet they also benefit by feedback from the trained classifier.

4.5. Time complexity for selecting one feature

Let n, M, and N denote the numbers of features, hidden neurons, and samples, respectively. Then the time complexity of R_{SM}FS method is $O(n^2M)$. The time complexity of wrapper approaches feature selection using RBFNN is $O(n(MnN + N^3))$ due to the training of n additional classifiers. The filter approaches (e.g., mutual information approaches) have a time complexity of $O(n^2N^2)$ due to the point-wise comparisons in computing the selection criterion. Moreover, R_{SM}FS method requires only one classifier training, while wrapper approaches require n classifier trainings. Usually, $M \leqslant N$ and the R_{SM}FS method is advantageous when the dataset consists of large numbers of samples and features.

4.6. Limitations of R_{SM}FS

The R_{SM}FS feature selection method depends on the ST-SM to find the error bound (R^*_{SM}) for the unseen samples located in the Q-union. So, the R_{SM}FS method cannot be applied to those classifiers in which their ST-SMs could not be defined (e.g., decision tree and rule-based systems). Moreover, as with all other wrapper methods, the feature subsets selected by the R_{SM}FS method are biased to the classifier. So, the feature subsets selected for RBFNN using R_{SM}FS method may not be suitable for SVM.

Although a local concept is adopted in the R^*_{SM}, a few portions of unseen samples located outside the Q-union will not have a big impact on the R_{SM}FS method [1]. However, in applications that depend heavily on detecting or recognizing outliers, both R^*_{SM} and R_{SM}FS may not be suitable choices.

5. Experimental results

We perform a thorough comparison among the proposed R_{SM}FS method and five other feature selection methods using the UCI Wine dataset [21] in Section 5.1 On top of the average testing accuracies of the RBFNNs built using the selected feature subsets by those methods, we also compare the features selected and show the advantages of the R_{SM}FS method. In Section 5.2, we provide more experimental results on a variety of datasets in terms of numbers of features samples and classes The methods being compared are Similarity [14], Separability [16], Correlation Coefficient [6], Mutual Information [13], and Leave-One-Out. These are compared with 5-fold CV. The maximum information compression index introduced in Ref. [14] will be used as the similarity measure.

In all our experiments, we perform 10 independent runs with random splitting of 50% of the samples to be the training dataset and the other samples to be the testing dataset. In all the experiments, we use the same RBFNN architecture and training procedures discussed at the end of Section 4.1. Samples in the testing dataset will not be used for the training; rather, they serve as unseen samples to test the generalization capability of the classifiers trained using the training dataset and selected feature subset. Thus, the 5-fold CV for the Leave-One-Out method will split the validation dataset from the training dataset rather than the testing dataset. This is because the unseen samples should be unknown when we train the classifier.

5.1. *Features selected by those feature selection methods*

We perform a thorough investigation of the six feature selection methods using the UCI Wine dataset. The UCI Wine dataset consists of 13 input features, 178 samples, and 3 classes. In all the experiments, we use seven hidden neurons for the RBFNN [1] and train them using the two-stage training method described in Section 4.1. Table 1 shows the three most relevant features selected by these methods. The feature subsets selected in different-random runs are different, and thus we select the feature subsets that occur in most of the random runs for each feature selection method.

From Table 2, one may notice that the average testing accuracies of the RBFNNs trained using the feature subsets selected by the R_{SM}FS method outperform all the others. When only 10 features are left for RBFNN training, the RBFNNs trained using the feature subset selected by the R_{SM}FS method yield the best average testing accuracy and outperform the one trained using the full set of features. The performances of Similarity and Leave-One-Out are the worst among all the methods. The Similarity method selects a feature subset based on the dissimilarity of a feature with respect to other features, regardless of whether it is relevant to the classification problem or not. Moreover, the similarity measure is computed using the statistical values of the features while ignoring the real distribution of individual samples. The idea of discarding similar features may be valid if we have a better similarity measure. However, the computational complexity of joint-distributions of features in datasets with large numbers of features is prohibitive. On the other hand, the Leave-One-Out method makes use of the estimation of generalization accuracy by the use of a validation dataset. However the training datasets consist of only 80% of the training samples and they may have too many differences from the original training dataset, especially for a small dataset like UCI Wine. In a later experiment, we find that this method performed better for large datasets (around 500,000 samples).

In Table 3, we compute the t-test values between the RBFNNs trained using the reduced feature subset and those trained using a full set of features. One may observe from Table 3 that the differences between the RBFNNs trained using a full set of features and those trained using only the two most relevant features selected by both R_{SM}FS and Mutual Information methods are statistically insignificant at the 0.05 level of significance (t-test value is smaller than 1.96 for approximately normally distributed testing accuracies). In contrast, the performances of the RBFNNs trained using the five most relevant features selected by the Leave-One-Out and Similarity methods and the RBFNNs trained using the three most relevant features selected by the Separability method are significantly worse than that of the RBFNNs trained using a full set of features.

In the following figures, we plot both the training and testing samples together so that we can visualize the generalization capability of the features being selected. In Fig. 5, we plot the two most relevant features selected by the Separability method. One may find that the class separability between the classes may not be the optimum and samples are mixed together in many regions. However, the

Table 1
The three most relevant features selected by the feature selection methods

Feature selection method	Most relevant feature	2nd most relevant feature	3rd most relevant feature
R_{SM}FS	13th Feature	12th Feature	7th Feature
MI	13th Feature	12th Feature	1st Feature
LOO	13th Feature	9th Feature	6th Feature
COR	12th Feature	7th Feature	6th Feature
SEPA	13th Feature	2nd Feature	1st Feature
SIM	10th Feature	2nd Feature	1st Feature

Table 3
t-Test values of the RBFNNs trained for the selected feature subsets

Feature selection method	Full set	10 Features	5 Features	3 Features	2 Features
R_{SM}FS	0	−0.51	−0.48	0.15	1.70
MI	0	−0.18	0.41	0.65	1.70
LOO	0	0.69	2.32	4.26	5.65
COR	0	0.57	−0.35	1.41	2.10
SEPA	0	−0.40	0.23	2.36	7.26
SIM	0	0.98	2.41	2.54	3.42

Table 2
Average (and standard deviation in parenthesis) testing accuracy for the selected feature subsets

Feature selection method	Full set	10 Features	5 Features	3 Features	2 Features
R_{SM}FS	92.57% (2.56%)	94.39% (2.52%)	94.27% (2.42%)	92.00% (2.81%)	85.30% (3.44%)
MI	92.57% (2.56%)	93.19% (2.36%)	90.07% (5.52%)	89.22% (4.47%)	85.30% (3.44%)
LOO	92.57% (2.56%)	88.76% (4.88%)	79.33% (5.09%)	65.12% (5.91%)	56.99% (5.75%)
COR	92.57% (2.56%)	90.47% (2.62%)	93.72% (2.05%)	85.60% (4.23%)	84.79% (2.68%)
SEPA	92.57% (2.56%)	94.05% (2.63%)	91.32% (4.70%)	77.23% (5.97%)	61.32% (3.46%)
SIM	92.57% (2.56%)	87.57% (4.39%)	77.80% (5.56%)	70.18% (8.43%)	56.89% (10.10%)

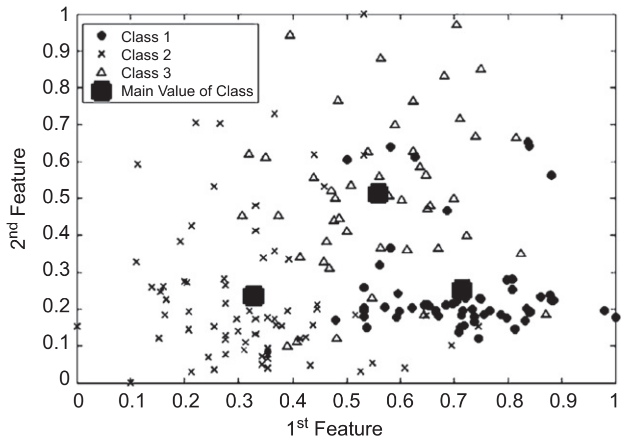

Fig. 5. Sample distribution of the two most relevant features selected by the Separability method for the UCI Wine dataset.

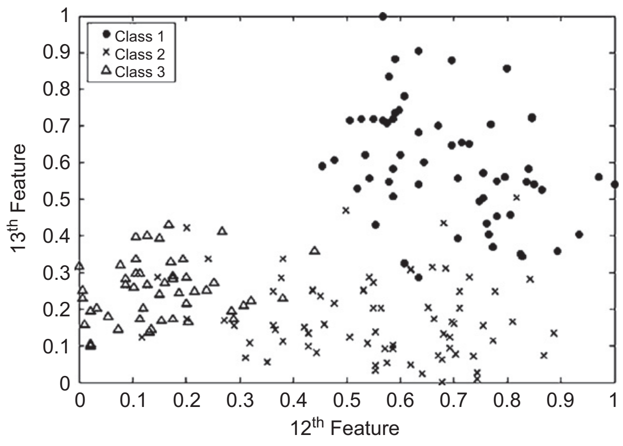

Fig. 7. Sample distribution of the two most relevant features selected by the Mutual Information and R_{SM}FS methods for the UCI Wine dataset.

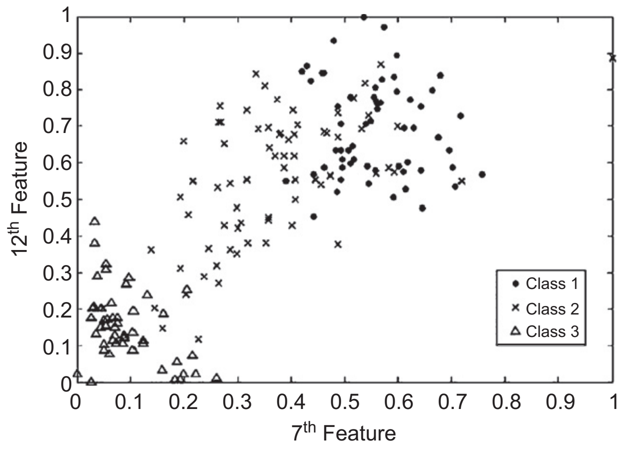

Fig. 6. Sample distribution of the two most relevant features selected by the Correlation Coefficient method for the UCI Wine dataset.

mean values, represented by big square dots in Fig. 5, of the three classes are separated far away from each other. This indicates that the separability measure is overly dependent on the mean values of the samples in each class.

Fig. 6 shows the two most relevant features selected by the Correlation Coefficients method. One may notice that these two features yield a very good linear correlation to the desired output because the class ID decreases from 3 to 1 when the values of these two features increase. However, the samples from Class 1 and Class 2 mix together and thus this feature subset yields a poor generalization capability.

Fig. 7 shows the two most relevant features (12th and 13th features) selected by both the Mutual Information and the R_{SM}FS methods. The desired outputs do not increase or decrease when the values of the 12th and 13th features increase (i.e. they do not have a linear correlation with the desired outputs). Although these two features do not show a good linear correlation to the output, they yield the best separability between the three classes among other choices of feature pairs. Thus, both methods select the feature subset with the best nonlinear correlation and separability to the output. Furthermore, we plot the sample distributions of the 3-feature subsets selected by the Mutual Information and the R_{SM}FS methods in Figs. 8 and 9, respectively. The 3-feature subset selected by the R_{SM}FS method yields a better separability of the classes than the one selected by the Mutual Information method. The Mutual Information method selects the feature subset based on the best nonlinear correlation to the output; thus it indirectly minimizes the training error during the feature selection. In contrast, the R_{SM}FS method minimizes the change in generalization error for the selection of feature subsets. Thus the feature subset removed by the R_{SM}FS method yields the least contribution to the approximation of the true unknown input–output mapping (i.e., the RBFNN f_θ), such that we could remove them with the least change in generalization performance. From Fig. 9, the generalization capability of the feature subset can be visualized as the separability of the samples from different classes and very few of them are mixed together. One may observe that the samples in Classes 2 and 3 have more overlapping in Fig. 8 than those in Fig. 9.

5.2. Experimental results for datasets with varieties in numbers of features and samples

In this section, we discuss the experiments on six more datasets in which we perform 10 independent runs for each, with 50% of the samples randomly selected as training datasets. A Gene dataset from Ref. [22] consists of extremely large numbers of features and small numbers of samples. The problem for the Gene dataset is to distinguish a tissue to be either carcinomatous or non-carcinomatous by its mRNA gene expression levels, with each feature corresponding to one mRNA gene expression. This is a typical bioinformatics dataset and is usually characterized by a large number of features with a small number of samples. In contrast, typical multimedia datasets consist of large numbers of features, samples, and classes. The Multiple Feature (MF) Digit Recognition and Isolated Letter (Isolet) Speech Recognition datasets are selected from the UCI machine learning repository [21]. They are typical multimedia datasets that consist of large numbers of features extracted from some transformations of the original speech signal or image. The large number of classes is also a characteristic of this kind of problem (e.g., 26 classes in English letter-recognition problems). On the other hand, datasets for network security problems usually consist of very large numbers of samples but medium numbers of features. The KDD CUP 99 dataset (Network Intrusion) is a typical network intrusion detection problem dataset that consists of a large number of samples but few features. Due to the growth of network usage and fast broadband Internet connections, the number of packets flowing through a server is very huge, and one is required to distinguish malicious and legitimate packets in an extremely short period of time. So, reducing the number of features for this type of problem is particularly important in lowering the response time of the intrusion detectors. We remove the samples in other attack types because their numbers

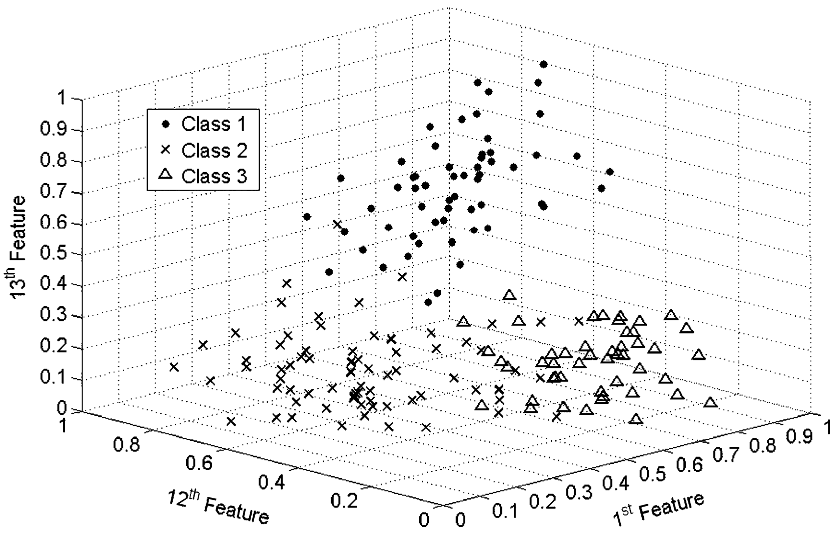

Fig. 8. Sample distribution of the three most relevant features selected by the Mutual Information method for the UCI Wine dataset.

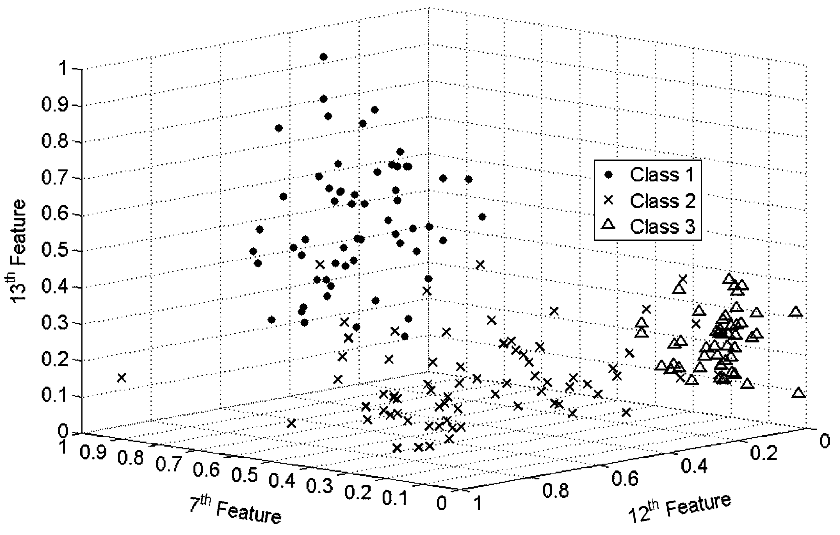

Fig. 9. Sample distribution of the three most relevant features selected by the R_{SM}FS method for the UCI Wine dataset.

Table 4
Characteristics of datasets and the number of hidden neurons used in experiments

Data set	Number of features	Number of samples	Number of classes	Number of hidden neurons in RBFNN
Gene	12,600	203	2	40
Isolet	617	6238	26	100
MF	649	2000	10	100
Network Intrusion	41	494,020	2	80
Sonar	60	208	2	23
Ionosphere	34	351	2	14

are too small when compared with the 494,020 normal and DoS attacks samples. Moreover, two typical medium-size machine learning datasets are selected from the UCI machine learning repository: Sonar and Ionosphere. In all of the experiments, we adopt the number of hidden neurons specified in Table 4 and therefore the performances of the RBFNNs are affected only by the feature subset being selected by these feature selection methods.

The Leave-One-Out method is infeasible for the Gene dataset, as it consists of 12,600 features, and it requires 396,931,500 RBFNNs to be trained for the 5-fold CV and combinations of the feature subsets. Another characteristic of typical bioinformatics datasets (e.g. the Gene dataset in Table 5) is that a huge number of features could be removed without a significant effect on the testing accuracies of the trained RBFNNs. The average testing accuracies using all methods are the same as those using a full set of features when the number of features is reduced from 12,600 to 500. The *t*-test values between the RBFNNs trained using a full set of features and those trained using reduced feature subsets are given in Table 5. From Table 5, the performance degradations of the RBFNNs trained using the 50 most relevant features selected by R_{SM}FS method are statistically insignificant, at the level of 0.05, when compared with those trained using a full set of features. In contrast, the performance degradations of the RBFNNs trained using the 50 most relevant features selected by all other methods are statistically significant. The average testing accuracy of Mutual Information with 50 features is better than the R_{SM}FS method; however, the overall trend of performance degradation from 12,600 features to 1 feature of the R_{SM}FS method is more superior.

The small number of samples distributed sparsely in such a high-dimensional space could be easily compressed to a lower-

Table 5
Average (standard deviation) testing accuracies and *t*-Test values of the RBFNNs trained using a subset of features for Gene dataset

# Features		R_{SM}FS	MI	COR	SIM	SEPA	LOO
12,600	Mean	97.03%	97.03%	97.03%	97.03%	97.03%	N/A
	Std Dev	1.29%	1.29%	1.29%	1.29%	1.29%	N/A
	t-Test	0.00	0.00	0.00	0.00	0.00	N/A
500	Mean	97.03%	97.03%	97.03%	97.03%	97.03%	N/A
	Std Dev	1.29%	1.29%	1.29%	1.29%	1.29%	N/A
	t-Test	0.00	0.00	0.00	0.00	0.00	N/A
250	Mean	99.01%	98.02%	91.09%	96.04%	97.03%	N/A
	Std Dev	2.28%	1.63%	2.80%	3.17%	1.29%	N/A
	t-Test	−0.76	−0.48	1.93	0.29	0.00	N/A
150	Mean	96.04%	90.83%	84.16%	85.85%	93.25%	N/A
	Std Dev	4.72%	4.89%	1.33%	2.09%	3.66%	N/A
	t-Test	0.20	1.23	6.95	4.55	0.97	N/A
50	Mean	90.10%	91.35%	82.18%	84.61%	86.50%	N/A
	Std Dev	3.33%	2.42%	1.74%	4.12%	4.64%	N/A
	t-Test	1.94	2.07	6.86	2.88	2.19	N/A
1	Mean	86.14%	82.18%	82.18%	82.18%	82.18%	N/A
	Std Dev	1.63%	2.10%	1.88%	4.73%	2.07%	N/A
	t-Test	5.24	4.42	6.51	3.03	6.09	N/A

Table 7
Average (standard deviation) testing accuracies and *t*-Test values of the RBFNNs trained using a subset of features for Multiple Feature Digit Recognition dataset

# Features		R_{SM}FS	MI	COR	SIM	SEPA	LOO
649 (100%)	Mean	96.90%	96.90%	96.90%	96.90%	96.90%	96.90%
	Std Dev	0.56%	0.56%	0.56%	0.56%	0.56%	0.56%
	t-Test	0.00	0.00	0.00	0.00	0.00	0.00
519 (80%)	Mean	96.88%	96.60%	96.97%	96.32%	96.12%	95.80%
	Std Dev	0.52%	0.64%	0.61%	0.42%	0.53%	0.26%
	t-Test	0.03	0.35	−0.08	0.83	1.01	1.78
389 (60%)	Mean	97.12%	96.55%	96.85%	95.16%	96.67%	94.79%
	Std Dev	0.49%	0.49%	0.65%	0.34%	0.55%	0.53%
	t-Test	−0.30	0.47	0.06	2.66	0.29	2.74
260 (40%)	Mean	96.45%	95.85%	95.76%	93.54%	96.56%	92.64%
	Std Dev	0.47%	0.46%	0.97%	0.67%	0.48%	0.46%
	t-Test	0.62	1.45	1.02	3.85	0.46	5.88
130 (20%)	Mean	97.42%	94.33%	92.93%	91.18%	94.62%	86.58%
	Std Dev	0.57%	0.61%	1.68%	1.57%	0.36%	0.70%
	t-Test	−0.65	3.10	2.24	3.43	3.42	11.51
65 (10%)	Mean	97.00%	93.95%	88.94%	85.88%	91.74%	82.29%
	Std Dev	0.57%	0.61%	1.68%	1.57%	0.36%	0.70%
	t-Test	−0.13	3.10	4.70	3.20	7.31	18.61

Table 6
Average (standard deviation) testing accuracies and *t*-Test values of the RBFNNs trained using a subset of features for isolated letter speech recognition dataset

# Features		R_{SM}FS	MI	COR	SIM	SEPA	LOO
617 (100%)	Mean	83.85%	83.85%	83.85%	83.85%	83.85%	83.85%
	Std Dev	0.99%	0.99%	0.99%	0.99%	0.99%	0.99%
	t-Test	0.00	0.00	0.00	0.00	0.00	0.00
494 (80%)	Mean	85.69%	84.52%	82.77%	82.31%	82.00%	82.56%
	Std Dev	2.21%	1.09%	1.66%	1.66%	1.16%	1.47%
	t-Test	−0.76	−0.46	0.56	0.80	1.21	0.73
370 (60%)	Mean	83.08%	82.52%	80.00%	76.31%	73.60%	77.08%
	Std Dev	1.24%	0.53%	2.54%	1.54%	2.01%	1.16%
	t-Test	0.49	1.18	1.41	4.12	4.57	4.44
247 (40%)	Mean	80.46%	77.45%	67.69%	71.54%	46.77%	68.61%
	Std Dev	1.66%	1.48%	1.05%	4.75%	1.63%	0.67%
	t-Test	1.75	3.59	11.20	2.54	19.44	12.75
123 (20%)	Mean	68.77%	63.52%	46.77%	62.00%	38.92%	43.30%
	Std Dev	1.40%	1.40%	1.79%	5.55%	1.40%	3.00%
	t-Test	8.79	11.86	18.13	3.88	26.20	12.84
62 (10%)	Mean	62.15%	38.78%	35.08%	48.92%	30.62%	30.65%
	Std Dev	4.02%	3.87%	4.94%	6.54%	4.44%	2.40%
	t-Test	5.24	11.28	9.68	5.28	11.70	20.49

Table 8
Average (standard deviation) testing accuracies and *t*-Test values of the RBFNNs trained using subset of features for Intrusion Detection dataset

# Features		R_{SM}FS	MI	COR	SIM	SEPA	LOO
41 (100%)	Mean	99.54%	99.54%	99.54%	99.54%	99.54%	99.54%
	Std Dev	0.06%	0.06%	0.06%	0.06%	0.06%	0.06%
	t-Test	0.00	0.00	0.00	0.00	0.00	0.00
33 (80%)	Mean	99.54%	99.41%	99.54%	99.49%	99.33%	99.54%
	Std Dev	0.06%	0.06%	0.06%	0.04%	0.10%	0.06%
	t-Test	0.00	1.53	0.00	0.69	1.80	0.00
25 (60%)	Mean	99.50%	99.41%	99.48%	98.39%	99.13%	99.42%
	Std Dev	0.06%	0.07%	0.08%	0.04%	0.16%	0.07%
	t-Test	0.47	1.41	0.60	15.95	2.40	1.30
16 (40%)	Mean	99.46%	99.22%	99.25%	93.44%	99.05%	99.41%
	Std Dev	0.09%	0.04%	0.06%	2.00%	0.12%	0.08%
	t-Test	0.74	4.44	3.42	3.05	3.65	1.30
8 (20%)	Mean	99.30%	99.18%	99.14%	92.14%	98.92%	99.51%
	Std Dev	0.13%	0.08%	0.09%	1.91%	0.10%	0.15%
	t-Test	1.68	3.60	3.70	3.87	5.32	0.19
4 (10%)	Mean	98.50%	98.43%	98.22%	80.51%	98.83%	99.51%
	Std Dev	0.18%	0.10%	0.08%	4.66%	0.13%	0.10%
	t-Test	5.48	9.52	13.20	4.08	5.17	0.77

dimensional space without affecting the separability between classes. Both the Mutual Information and R_{SM}FS methods find feature subsets that yield RBFNNs with better average testing accuracies than those trained using a full set of features.

From Table 6, one may notice that the Separability method performs very poorly because of the large number of classes in the problem, since samples from the 26 classes are mixed together. However, from Table 7, one can see that the Separability method performance is reasonable and this indicates that the performance of the Separability method is dataset dependent. This is because the separability measure usually makes use of the mean values of the samples in each class, which is not suitable for datasets in which the samples in a class surround the samples in another class. On the other hand, the major drawback of the Leave-One-Out method is the reservation of validation datasets and this changes the training datasets. So, it performs relatively poorly in most of the datasets except the Intrusion Detection dataset (see Table 8). The Correlation method also works poorly for the Isolet dataset (see Table 6) and this may be because of the nonlinear relationship between the input features and outputs being more important than the linear one in this dataset, due to the large number of classes More importantly, from Tables 6 and 7, the proposed method removes 60% and 90% of the features without causing statistically significant loss of the RBFNNs' performances, respectively, for the experiments of the Isolet Speech Recognition and the MF Digit Recognition datasets.

One may notice from Table 8 that the testing accuracies for all methods have insignificant differences in their percentages. However, every 0.01% of the testing accuracy represents 23 samples because of the large number of testing samples (247,010). In particular, the Leave-One-Out method performs very well in this dataset due to the large number of training samples and this reduces the differences in training sets caused by the reservation of validation sets for the 5-fold CV. In contrast, the Similarity method performs the worst in the Intrusion Detection dataset because the statistically

Table 9
Average (standard deviation) testing accuracies and *t*-Test values of the RBFNNs trained using a subset of features for Sonar dataset

# Features		R_{SM}FS	MI	COR	SIM	SEPA	LOO
60 (100%)	Mean	83.04%	83.04%	83.04%	83.04%	83.04%	83.04%
	Std Dev	2.02%	2.02%	2.02%	2.02%	2.02%	2.02%
	t-Test	0	0	0	0	0	0
48 (80%)	Mean	82.47%	81.05%	80.37%	82.89%	81.15%	82.12%
	Std Dev	2.47%	3.21%	3.04%	1.86%	3.44%	2.78%
	t-Test	0.18	0.52	0.73	0.05	0.47	0.27
36 (60%)	Mean	82.38%	80.95%	80.66%	81.99%	81.53%	80.67%
	Std Dev	3.39%	3.21%	3.46%	3.68%	2.63%	2.06%
	t-Test	0.17	0.55	0.59	0.25	0.46	0.82
24 (40%)	Mean	81.59%	80.18%	81.34%	80.27%	81.34%	79.88%
	Std Dev	4.40%	3.99%	3.47%	3.76%	3.48%	3.60%
	t-Test	0.30	0.64	0.42	0.65	0.42	0.77
12 (20%)	Mean	79.09%	80.89%	77.65%	77.26%	78.72%	77.17%
	Std Dev	3.80%	3.85%	3.16%	3.61%	3.91%	3.02%
	t-Test	0.92	0.49	1.44	1.40	0.98	1.62
6 (10%)	Mean	75.45%	74.06%	76.39%	73.28%	77.07%	76.68%
	Std Dev	3.54%	3.41%	3.08%	3.13%	3.02%	3.86%
	t-Test	1.86	2.27	1.81	2.62	1.64	1.46

Table 10
Average (standard deviation) testing accuracies and *t*-Test values of the RBFNNs trained using a subset of features for Ionosphere dataset

# Features		R_{SM}FS	MI	COR	SIM	SEPA	LOO
34 (100%)	Mean	84.57%	84.57%	84.57%	84.57%	84.57%	84.57%
	Std Dev	0.73%	0.73%	0.73%	0.73%	0.73%	0.73%
	t-Test	0	0	0	0	0	0
27 (80%)	Mean	84.34%	84.34%	83.97%	83.54%	84.34%	82.43%
	Std Dev	1.25%	1.51%	1.52%	1.18%	0.74%	0.98%
	t-Test	0.16	0.14	0.36	0.74	0.22	1.75
20 (60%)	Mean	85.20%	85.03%	84.71%	84.23%	84.29%	81.29%
	Std Dev	2.00%	1.50%	1.53%	1.44%	1.94%	2.35%
	t-Test	−0.30	−0.28	−0.08	0.21	0.14	1.33
14 (40%)	Mean	85.03%	85.03%	84.70%	84.11%	84.80%	82.14%
	Std Dev	2.29%	1.68%	1.76%	2.10%	1.27%	2.00%
	t-Test	−0.19	−0.25	−0.07	0.21	−0.16	1.14
7 (20%)	Mean	86.06%	86.06%	85.20%	76.97%	75.43%	82.43%
	Std Dev	1.96%	1.91%	1.46%	2.11%	1.51%	2.83%
	t-Test	−0.71	−0.73	−0.39	3.40	5.45	0.73
3 (10%)	Mean	86.97%	84.63%	83.09%	74.80%	77.89%	82.57%
	Std Dev	2.12%	1.98%	2.06%	2.60%	1.40%	2.07%
	t-Test	−1.07	−0.03	0.68	3.62	4.23	0.91

similar features may have significant differences in individual samples and separability to the classes. Furthermore, Table 8 shows that the Leave-One-Out method removes 90% of features without making a statistically significant change to the RBFNN's performances while the R_{SM}FS removes only 80% of features in this experiment.

From Tables 9 and 10, the testing accuracies of the RBFNNs decrease significantly. However, owing to the small number of samples, most of the reductions in accuracies are not statistically significant. In contrast, one may observe from Table 10 that the RBFNNs trained using the three most relevant features selected by R_{SM}FS yield the highest average testing accuracy in the experiments of the Ionosphere dataset. In the experiments of Sonar Target, the performance degradation of all the feature selection methods is statistically insignificant when 80% of the features are removed.

One may notice from Table 11 that the numbers of classifiers required to be trained are the same for all methods except the Leave-One-Out method. Thus, for a dataset consisting of n features, it has to train $(n^2+n)/2$ classifiers and this is infeasible for a large dataset (e.g. the Gene dataset). Table 12 shows the average computational time for the feature selection process together with the trainings of all classifiers that produce the testing accuracies for each feature subset selected. The R_{SM}FS uses the least computational time in all experiments. The proposed method uses only 60% of the computational time when compared with other methods in experiments using the Sonar and Ionosphere datasets. The RBFNN training time dominates the computational time when the numbers of samples and features increase. In addition, the computational time is very large when the dataset consists of large numbers of features due to the one-by-one feature selection adopted in this work. The R_{SM}FS can feasibly remove more than one feature in each iteration. However, the computational effort in generating the feature subsets may become large, and the selection of the number of features being removed in each iteration needs further investigation.

Table 11
Number of trained classifiers required

Data set	.R_{SM}FS	MI	COR	SIM	SEPA	LOO
Gene	12,600	12,600	12,600	12,600	12,600	396,931,500
Isolet	617	617	617	617	617	190,653
MF	649	649	649	649	649	210,925
Network Intrusion	41	41	41	41	41	861
Sonar	60	60	60	60	60	1830
Ionosphere	34	34	34	34	34	595

Table 12
Average computational time for training all the RBFNN and feature selections

Data set	R_{SM}FS	MI	COR	SIM	SEPA	LOO
Gene	35 h	38 h	35 h	36 h	35 h	N/A
Isolet	2.5 h	3.0 h	2.5 h	2.5 h	2.5 h	76 h
MF	1.5 h	2.1 h	1.5 h	1.5 h	1.5 h	52 h
Network Intrusion	1.4 h	1.9 h	1.4 h	1.4 h	1.4 h	31.9 h
Sonar	9 s	20 s	17 s	15 s	19 s	588 s
Ionosphere	3 s	5 s	5 s	5 s	5 s	46 s

To summarize the experimental results, the R_{SM}FS is able to remove a large percentage of features without statistically significant decreases in the performances of the RBFNNs trained using the reduced feature subsets. The proposed method performs the best for all datasets in our experiments that consist of combinations of large and small numbers of features, samples and classes. In addition, the R_{SM}FS uses the least computational time and performs particularly well in experiments of datasets with large numbers of classes and features, while the number of samples does not have a significant effect on the R_{SM}FS.

The Mutual Information method ignores information from the trained classifiers; thus it is worse than the R_{SM}FS but still better than other methods in our experiments. Furthermore, it performs worse when the number of features is large. On the other hand, although the Leave-One-Out method makes use of the generalization performance to evaluate the feature subset, it requires reserving sets for validation and this may change the training dataset significantly when the K is small for the K-fold cross-validation. However, the Leave-One-Out method is very time consuming already for $K=5$; it will therefore become infeasible if we increase the value of K further.

5.3. An application to bankruptcy prediction

Bankruptcy and financial distress cost banks and businesses billions of dollars each year. It is not surprising, therefore, that credit rating agencies have devoted substantial resources to credit appraisal [23,24]. Methods used to predict the financial solvency of businesses range from statistical models [25] to approaches that involve

Table 13
Feature set of the bankruptcy dataset

−1 Year	−2 Year	−3 Year	Feature name
1	43	85	Total assets
2	44	86	Working capital
3	45	87	Cash
4	46	88	Marketable securities
5@	47	89	Current ratio
6@	48	90	Acid test ratio
7	49	91	Quick assets
8	50	92	Current liabilities
9	51	93	Retained earning
10	52	94	Total debt
11	53	95	Long term debt
12	54	96	Shareholder's equity
13	55	97	Total debt/total capital
14	56	98	Total liabilities
15	57	99	Market value of equity
16$	58	100	Net profit margin
17$	59	101	Operating margin before depreciation
18@	60	102	Pre-tax profit margin
19	61	103	Interest coverage before tax
20	62	104	Sales
21	63	105	Inventory turnover
22	64	106	Inventory
23	65	107	Cost goods sold
24@	66	108@	Receivable turnover
25@	67	109	Account receivable
26	68	110	Earning before interest and taxes
27	69	111	Net income
28	70	112	Cash flow
29	71	113	Cash flow from operation
30	72	114	Current asset
31	73	115	Net plant
32*	74@	116	Sales growth in previous 3 years
33$	75$	117$	Price end
34	76$	118$	Dividend
35*	77*	119*	Stock return
36$	78$	120$	Market return
37*	79*	121*	LERET
38	80	122	Market value of firm
39$	81$	123$	Share price
40	82	124	Number of outstanding shares
41	83	125	M/B ratio
42	84	126$	LSIGMA = std_dev(monthly price close at that year)

@ Indicate the 22nd–28th irrelevant features. $ Indicate the 8th–21st irrelevant features. ∗ Indicate the first seven most irrelevant features.

Table 14
Averages and standard deviations of training and testing accuracies of the RBFNNs trained using a subset of features for the bankruptcy dataset

# Features		R_{SM}FS	MI	SEPA
126 (100%)	Mean	58.40%	58.40%	58.40%
	Std Dev	6.23%	6.23%	6.23%
	t-Test	0	0	0
119 (94%)	Mean	60.06%	55.13%	52.41%
	Std Dev	7.75%	11.13%	5.82%
	t-Test	−0.12	0.19	0.50
105 (83%)	Mean	58.59%	50.96%	47.76%
	Std Dev	5.92%	7.34%	2.89%
	t-Test	−0.02	0.55	1.17
99 (79%)	Mean	57.24%	53.37%	47.44%
	Std Dev	6.16%	10.54%	3.18%
	t-Test	0.09	0.30	1.16
76 (60%)	Mean	47.88%	51.28%	50.80%
	Std Dev	7.94%	5.66%	9.56%
	t-Test	0.74	0.60	0.48
50 (40%)	Mean	41.79%	46.80%	45.67%
	Std Dev	5.66%	7.08%	5.77%
	t-Test	1.40	0.87	1.06
25 (20%)	Mean	39.04%	37.82%	36.54%
	Std Dev	5.92%	7.20%	6.31%
	t-Test	1.59	1.53	1.74
13 (10%)	Mean	30.77%	30.29%	25.25%
	Std Dev	6.88%	9.83%	5.06%
	t-Test	2.11	1.75	2.94

extensive subjective human judgment. There are a lot of data available to analysts from which they can predict bankruptcy but the choice of which variables to use is often ad hoc. The large number of variables and the large universe of firms make our R_{SM}FS methodology particularly suitable as a tool for predicting bankruptcy.

To assess the merits of R_{SM}FS in the prediction of bankruptcy we use a large sample of firms and financial variables drawn from US corporations in the COMPUSTAT database. We are able to identify a sample of 312 firms that filed for bankruptcy in the period 1986–2005. We also draw a sample of 9431 firms that had no bankruptcy. Therefore, in our test sample we have 3.2% of firms that are bankrupt and 96.8% of firms that remain solvent and these percentages reflect the relatively infrequent nature of bankruptcies. Note, however, that when bankruptcies do occur they result in huge financial losses for creditors, bankers, and investors. We make use of 42 accounting and financial variables and calculate these variables at years −1, −2, and −3 relative to the year of bankruptcy (or non-bankruptcy for the solvent firms); this yields 126 features. The features consist of financial variables that are potential indicators of a firm's financial health [25,26]. The features are listed in Table 13.

The total sample of 9743 company observations are divided into a training set (4872 observations) and a testing set (4871 observations) drawn randomly in 10-folds. Both the training and testing datasets therefore have 156 bankrupt companies and 4716 solvent companies. The emphasis in our study is on the correct classification of bankrupt companies since the misclassification of bankrupt companies as solvent companies results in substantial losses to investors and lenders. If a company is predicted to be solvent, then banks will lend money to it and investors will buy shares in it. If the company then becomes bankrupt, bankers and investors will lose their money. In contrast, the costs of misclassifying a solvent firm as bankrupt are relatively small and may be close to zero when banks and investors have many lending and investment opportunities.

The testing accuracies of the R_{SM}FS, Mutual Information and Separability methods are given in Table 14. The testing accuracy of the full set of 126 features is 58.40%. In this experiment, we only report the testing accuracy of the prediction in bankrupt companies. Owing to the imbalance of solvent (96.8%) and bankrupt (3.2%) companies, 58.40% testing accuracy on the bankrupt companies' prediction should be considered as reasonable. A reduced set of 119 features yields the highest accuracy rate of 60.06%. Further reducing the feature set to 105 features gives accuracy rates of 58.59%, which still performs better than the one using a full set of features. Thus the R_{SM}FS method is able to meaningfully reduce the set of features for predicting bankruptcy without unduly reducing accuracy. However, owing to the nature of the problem that a small drop in testing accuracy could cause large investment losses, we are interested only in those feature subsets that produce an accuracy performance at least as good as the one by using the full set of features. The discussion on statistical significance (or insignificance) for this problem may not be meaningful due to the very large standard deviations, which is caused by the small bankrupt samples.

Table 13 highlights those variables that contribute least to the testing accuracy. The cells marked with ∗ represent the first 7 features removed; these are the least relevant. The cells marked with $ represent the next 14 features removed (the next least relevant features) and the cells marked with @ the next 7 features removed.

One striking feature of Table 13 is that many of the removed features are from year −1. At first glance this might appear counter-intuitive. That is, the most recent data are the least relevant. Our explanation for this is that some companies that are in financial distress resort to manipulating or massaging their financial statements and so the reported numbers are distorted or even downright fraudulent [27]. Recent corporate scandals (e.g., Enron, Worldcom) have highlighted the ways in which management manipulates financial statements [27,28]. The results indicate that the financial statements in year −2 and year −3 reflect the deteriorating position of the soon-to-be bankrupt companies but management has not yet started to manipulate the accounts in a major way.

Another striking feature of Table 13 is that stock market prices (price end, stock return, and market return) and stock market risk (LERET and LSIGMA) are largely irrelevant in predicting bankruptcy. Although one can argue that stock market variables should capture fundamental information about a company's prospects and risk [25], our results for the prediction of bankruptcy provide little support for this viewpoint. The ability of stock market variables to reflect fundamental economic factors has also come under recent attack by regulators and financial commentators. For example, Alan Greenspan, the ex-Chairman of the US Federal Reserve, coined the term 'irrational exuberance' to describe what he saw as an over-priced stock market. As another example, the financial scandals of the late 1990s and early 2000s and the ongoing sub-prime banking catastrophes of 2007 and 2008 were largely unpredicted by the stock markets until it was too late. Similarly, the internet stock bubble in the US was a classic example of over-valuation by the stock market. In light of this evidence perhaps we should not be too surprised that stock market variables do not do a good job in predicting bankruptcies [29].

Our analysis indicates that R_{SM}FS is a valuable tool in developing a prediction model for corporate bankruptcies. It is able to substantially reduce the number of features without a significant loss of testing accuracy for unseen samples. This suggests that the method will be very useful for financial and credit analysts.

5.4. Experimental comparisons with Gaussian kernel SVM

When compared with SVM, RBFNN is more suitable for problems with a large number of training samples (N) and multi-classes because SVM uses nonlinear optimization of its N-by-N kernel matrix. However, when one is given a two-class problem, an interesting question arises whether should one use RBFNN or SVM for solving it. In Section 5.4.1, we compare R_{SM}FS with the feature selection using the R2W2 generalization error bound of a Gaussian kernel SVM [30]. In Section 5.4.2, we replace the k-means clustering algorithm by SVM with Gaussian kernel for center selection in RBFNN training [31].

5.4.1. Comparison with feature selection by R2W2 error bound of SVM

In Ref. [30], the authors proposed a feature selection method for SVM based on its R2W2 error bound. We perform experiments of this method on three 2-class datasets, Gene, Wine and Sonar. In Tables 15–17, the testing accuracies of RBFNN trained using reduced features selected by R_{SM}FS, for these three datasets are compared with Gaussian SVM trained using reduced features selected by the R2W2 error bound. The t-test value for SVM is computed with respect to the testing accuracy of SVM trained with a full set of features and shows the statistical significance of the reduction in SVM testing accuracy caused by the R2W2 error bound feature reduction.

SVM outperforms RBFNN when all 12,600 features are used to train the classifiers. However, the difference in the performances of these two classifiers is statistically insignificant. More importantly, RBFNN trained with features selected by R_{SM}FS achieves the best average testing accuracy when there are only 250 features left. In contrast, the testing accuracies of SVM trained using features selected by R2W2 error bound fall by a statistically significant amount (t-test value larger than 1.96) when there are 500 features left.

From Table 16, SVM outperforms RBFNN when a full set of features is adopted. However, the average testing accuracy falls when features are removed by the R2W2 error bound. One may notice that the RBFNN trained using 3 (10%) features selected by R_{SM}FS outperforms the SVM trained using 27 (80%) features selected by R2W2 error bound by about 0.8% average testing accuracy.

From Table 17, the average testing accuracy of SVM falls a statistically significantly amount when there are only 12 (20%) features left for training. In contrast, RBFNN with R_{SM}FS yields a statistically insignificant fall in average testing accuracy even when there are only 6 (10%) features left for training.

To summarize, SVM with Gaussian kernel outperforms RBFNN when a full set of features is used for training. However, when SVM works with the R2W2 error bound feature selection, the performances of SVM are worse than RBFNN with feature selection by R_{SM}FS for smaller feature subsets. One possible reason for this is

Table 15
Average (standard deviation) testing accuracies and t-Test values for Gene dataset

# Features		RBFNN with R_{SM}FS	Gaussian SVM with R2W2
12,600	Mean	97.03%	97.20%
	Std Dev	1.29%	0.40%
	t-Test	0.00	0.00
500	Mean	97.03%	91.75%
	Std Dev	1.29%	2.31%
	t-Test	0.00	2.00
250	Mean	99.01%	88.94%
	Std Dev	2.28%	1.32%
	t-Test	−0.76	4.80
150	Mean	96.04%	87.29%
	Std Dev	4.72%	1.59%
	t-Test	0.20	4.97
50	Mean	90.10%	88.45%
	Std Dev	3.33%	1.62%
	t-Test	1.94	4.33
1	Mean	86.14%	73.60%
	Std Dev	1.63%	1.62%
	t-Test	5.24	6.97

Table 16
Average (standard deviation) testing accuracies and t-Test values for Ionosphere dataset

# Features		RBFNN with R_{SM}FS	Gaussian SVM with R2W2
34 (100%)	Mean	84.57%	89.61%
	Std Dev	0.73%	2.18%
	t-Test	0.00	0.00
27 (80%)	Mean	84.34%	86.18%
	Std Dev	1.25%	2.93%
	t-Test	0.16	0.67
20 (60%)	Mean	85.20%	84.68%
	Std Dev	2.00%	3.46%
	t-Test	−0.30	0.87
14 (40%)	Mean	85.03%	82.81%
	Std Dev	2.29%	3.35%
	t-Test	−0.19	1.23
7 (20%)	Mean	86.06%	79.53%
	Std Dev	1.96%	3.27%
	t-Test	−0.71	1.85
3 (10%)	Mean	86.97%	79.01%
	Std Dev	2.12%	3.92%
	t-Test	−1.07	1.74

Table 17
Average (standard deviation) testing accuracies and *t*-Test values for Sonar dataset

# Features		RBFNN with R_{SM}FS	Gaussian SVM with R2W2
60 (100%)	Mean	83.04%	84.76%
	Std Dev	2.02%	2.00%
	t-Test	0	0.00
48 (80%)	Mean	82.47%	84.47%
	Std Dev	2.47%	2.59%
	t-Test	0.18	0.06
36 (60%)	Mean	82.38%	82.04%
	Std Dev	3.39%	3.67%
	t-Test	0.17	0.48
24 (40%)	Mean	81.59%	73.79%
	Std Dev	4.40%	5.22%
	t-Test	0.30	1.52
12 (20%)	Mean	79.09%	59.90%
	Std Dev	3.80%	4.25%
	t-Test	0.92	3.98
6 (10%)	Mean	75.45%	56.60%
	Std Dev	3.54%	5.36%
	t-Test	1.86	3.83

that the decision boundary of the classification problem becomes more complicated when the number of features is reduced, but SVM still attempts to find a separating hyperplane that separates samples in the two classes, which may make SVM overfit. Another issue is that the R2W2 error bound may not make full use of information of the training samples while the R_{SM} (localized generalization error) makes use of statistical information of each feature in the input space. Further investigation may be needed on the theoretical comparison between these two error bounds. From current experimental results, R_{SM}FS performs better.

5.4.2. SVM initialization of centers for RBFNN

The R_{SM}FS and R_{SM} make use of parameter values of the resulting RBFNN only and regardless of how the RBFNNs are trained. In previous sections, we train RBFNN with one of the most standard methods. In this section, we adopt SVM initialization to find the centers of RBFNN [31] instead of using *k*-means. We compare the R_{SM}FS with two other feature selection methods described in previous experiments: Mutual Information and Separability methods. Experimental results of Sonar and Ionosphere datasets are presented in Tables 18 and 19, respectively.

Our experimental results using a full set of features coincide with the results in Ref. [31] in that RBFNN performs better when the RBFNN centers are selected by SVM instead of *k*-means. However, when the percentage of features used for training becomes small (40% or less), the performances of RBFNN initialized by *k*-means perform better. This may be due to the nature of the two RBFNN center initialization methods. For SVM initialization, support vectors on the decision boundary are used as the RBFNN centers. As shown in Refs. [31,32], support vectors of SVM with Gaussian kernel surround the cluster of samples in the input space. In *k*-means, centroids of sample clusters are found for the RBFNN centers.

Nonetheless, the R_{SM}FS is able to make a large reduction in the number of features without a statistically significant change to the testing accuracy of the RBFNNs for both *k*-mean and SVM initialization of RBFNN centers. Similar to the previous experimental results, the R_{SM}FS selects better feature subsets in the sense that the RBFNNs trained with it perform better than those trained using feature subsets selected by Mutual Information and Separability. This indicates that the R_{SM}FS is robust to changes in the initialization and training methods of RBFNN.

Table 18
Average (standard deviation) testing accuracies and *t*-Test values of the RBFNNs (with SVM initialization of centers) trained using a subset of features for Sonar dataset

# Features		R_{SM}FS	MI	SEPA
60 (100%)	Mean	85.63%	85.63%	85.63%
	Std Dev	1.87%	1.87%	1.87%
	t-Test	0	0	0
48 (80%)	Mean	85.63%	84.47%	80.97%
	Std Dev	1.87%	2.28%	2.96%
	t-Test	0	0.28	0.47
36 (60%)	Mean	85.24%	83.11%	80.19%
	Std Dev	0.43%	2.80%	3.47%
	t-Test	0.17	0.54	1.02
24 (40%)	Mean	79.61%	79.81%	78.25%
	Std Dev	4.50%	2.32%	4.64%
	t-Test	0.95	1.39	1.13
12 (20%)	Mean	79.81%	74.56	75.73%
	Std Dev	3.32%	3.02%	2.99%
	t-Test	1.12	2.26	2.04
6 (10%)	Mean	70.87%	65.83%	67.18%
	Std Dev	3.88%	6.00%	6.23%
	t-Test	2.57	2.52	2.28

Table 19
Average (standard deviation) testing accuracies and *t*-Test values of the RBFNNs (with SVM initialization of centers) trained using a subset of features for Ionosphere dataset

# Features		R_{SM}FS	MI	SEPA
34 (100%)	Mean	86.61%	86.61%	86.61%
	Std Dev	1.71%	1.71%	1.71%
	t-Test	0	0	0
27 (80%)	Mean	86.94%	85.14%	86.61%
	Std Dev	1.76%	3.15%	1.95%
	t-Test	−0.09	0.30	0.31
20 (60%)	Mean	88.49%	84.82%	86.20%
	Std Dev	2.27%	4.15%	1.91%
	t-Test	−0.47	0.31	0.11
14 (40%)	Mean	80.90%	80.90%	80.08%
	Std Dev	5.84%	7.41%	5.70%
	t-Test	0.76	0.63	0.88
7 (20%)	Mean	77.96%	76.41%	77.80%
	Std Dev	2.78%	4.73%	1.66%
	t-Test	1.93	1.58	2.61
3 (10%)	Mean	86.45%	82.78%	83.10%
	Std Dev	1.39%	3.68%	0.86%
	t-Test	0.05	0.71	1.36

6. Conclusion and discussion

In this paper, we propose removing features that do not contribute to the localized generalization error bound (R_{SM}^*) of the classifier. The R_{SM}^* model bounds from above the generalization error of unseen samples located within a neighborhood of the training samples. In the experiments for two of the datasets, the RBFNNs built using feature subsets with 90% of features removed by the R_{SM}FS yield average testing accuracies higher than those trained using a full set of features. Moreover, the experimental results show that the proposed method consistently removes large percentages of features with a statistically insignificant loss of testing accuracy for unseen samples. Experimental results also show that the R_{SM}FS is robust to changes in initialization and training methods of RBFNN. In addition, the R_{SM}FS is fast and independent of the number of samples.

Optimal feature subset selection with low time complexity is still an open problem in feature selection. Fortunately, the computation of the proposed R^*_{SM} has a low time complexity. This may provide research opportunities on low complexity feature selection methods, and this is an important future research direction for us. Furthermore, we assume the training dataset to be representative of the classification problem that we want to solve. However this may not be the case in some situations. Extensions of the current work to adopt incomplete information or temporal dynamic problems are also important and challenging works.

Furthermore, one may notice that the R^*_{SM} is applicable to other types of classifiers by replacing the ST-SM term (e.g., multilayer perceptron neural networks, support vector machines, etc.). One feasible future research topic is to select both the classifier type and feature subset by comparing the differences among their R^*_{SM} values for a given dataset. This would be particularly useful to practical problem solving when a large number of features and several choices of classifiers are available.

Acknowledgment

This work is supported by the Hong Kong Polytechnic University Research Grant G-YD87.

References

[1] D.S. Yeung, W.W.Y. Ng, D. Wang, E.C.C. Tsang, X-Z. Wang, Localized generalization error and its application to architecture selection for radial basis function neural network, IEEE Trans. Neural Networks (2007) 1294–1305.
[2] W.W.Y. Ng, A. Dorado, D.S. Yeung, W. Pedrycz, E. Izquierdo, Image classification with the use of radial basis function neural networks and the minimization of localized generalization error, Pattern Recognition (2007) 19–32.
[3] S. Haykin, Neural Networks, Prentice-Hall, Englewood Cliffs, NJ, 1998.
[4] J. Moody, C.J. Darken, Fast learning in networks of locally-tuned processing units, Neural Comput. (1989) 281–294.
[5] L. Blum, P. Langley, Selection of relevant features and examples in machine learning, Artif. Intell. (1997) 245–271.
[6] H. Liu, H. Motoda, Feature Selection for Knowledge Discovery and Data Mining, Kluwer Academic Publisher, Dordrecht, 1998.
[7] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, J. Mach. Learn. Res. (2003) 1157–1182.
[8] J.R. Quinlan, Decision trees and decision-making, IEEE Trans. SMC (1990) 339 –346.
[9] I.T. Jolliffe, Principal Component Analysis, second ed., Springer, Berlin, 2002.
[10] A. Malhi, R.X. Gao, PCA-based feature selection scheme for machine defect classification, IEEE Trans. Instrum. Meas. (2004) 1517–1525.
[11] R. Battiti, Using mutual information for selecting features in supervised neural net learning, IEEE Trans. Neural Networks (1994) 537–550.
[12] H. Peng, F. Long, C. Ding, Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy, IEEE Trans. PAMI (2005) 1226–1238.
[13] N. Kwak, C.-H. Choi, Input feature selection by mutual information based on parzen window, IEEE Trans. PAMI (2002) 1667–1671.
[14] P. Mitra, C.A. Murthy, S.K. Pal, Unsupervised feature selection using feature similarity, IEEE Trans. PAMI (2002) 301–312.
[15] K.Z. Mao, Orthogonal forward selection and backward elimination algorithms for feature subset selection, IEEE Trans. SMC (Part B) (2004) 629–634.
[16] X. Fu, L. Wang, Data dimensionality reduction with application to simplifying RBF network structure and improving classification performance, IEEE Trans. SMC (Part B) (2003) 399–409.
[17] G.V. Lashkia, L. Anthony, Relevant, irredundant feature selection and noisy example elimination, IEEE Trans. SMC (Part B) (2004) 888–897.
[18] R. Kohavi, G.H. John, Wrappers for feature subset selection, Artif. Intell. (1997) 273–324.
[19] J. Bi, K.P. Bennett, M. Embrechts, C.M. Breneman, M. Song, Dimensionality reduction via sparse support vector machines, J. Mach. Learn. Res. (2003) 1229–1243.
[20] T. Hastie, R. Tibshirani, J. Friedman, The Element of Statistical Learning, Springer, Berlin, 2001.
[21] UCI Machine Learning Repository, ⟨http://archive.ics.uci.edu/ml/⟩.
[22] A. Bhattacharjee, W.G. Richards, J. Staunton, C. Li, S. Monti, P. Vasa, C. Ladd, J. Beheshti, R. Bueno, M. Gillette, M. Loda, G. Weber, E.J. Mark, E.S. Lander, W. Wong, B.E. Johnson, T.R. Golub, D.J. Sugarbaker, M. Meyerson, Classification of human lung carcinomas by mRNA expression profiling reveals distinct adenocarcinoma subclasses, Proc. Natl. Acad. Sci. USA (2001) 13790–13795.
[23] D. Duffie, K. Singleton, Credit Risk: Pricing, Measurement and Management, Princeton University Press, Princeton, 2003.
[24] W. Poon, M. Firth, Are unsolicited ratings lower? International evidence from bank ratings, J. Bus. Finance Accounting (2005) 1741–1771.
[25] E. Altman, Bankruptcy, Credit Risk, and High Yield Junk Bonds, Blackwell, Oxford, 2001.
[26] W.H. Beaver, M.F. McNichols, J.W. Rhie, Have financial statements become less informative? Evidence from the ability of financial ratios to predict bankruptcy, Rev. Accounting Stud. (2005) 93–122.
[27] P. Dechow, C. Schrand, Earnings Quality, Monograph, Research Foundation of the CFA Institute, 2004.
[28] G.J. Benston, A.L. Hartgraves, Enron: what happened and what we can learn from it, J. Accounting Public Policy (2002) 105–127.
[29] V. Agarwal, R. Taffler, Comparing the performance of market-based and accounting-based bankruptcy prediction models, J. Banking Finance, 2008, to appear.
[30] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, V. Vapnik, Feature selection for SVMs, Adv. Neural Inf. Process. Syst. (2001) 668–674.
[31] B. Scholkopf, K.K. Sung, C.J.C. Burges, F. Girosi, P. Niyogi, T. Poggio, V. Vapnik, Comparing support vector machines with Gaussian kernels to radial basis function classifiers, IEEE Trans. Signal Processing (1997) 2758–2765.
[32] W.W.Y. Ng, D.S. Yeung, D. Wang, E.C.C. Tsang, X-Z. Wang, Localized generalization error of Gaussian based classifiers and visualization of decision boundaries, Soft Comput. (2007) 375–381.

About the Author—WING W.Y. NG received the B.Sc. degree in Information Technology and the Ph.D. degree from Hong Kong Polytechnic University in 2001 and 2006, respectively. In 2008, he joined the School of Computer Science and Engineering, South China University of Technology, China, where he is currently an Associate Professor.

About the Author—DANIEL S. YEUNG received the Ph.D. degree in Applied Mathematics from Case Western Reserve University in 1974. He was the Chairman of the Department of Computing, The Hong Kong Polytechnic University, Hong Kong. He has been elected as President in 2007 for the IEEE SMC Society.

About the Author—MICHAEL FIRTH earned the Ph.D. from the University of Bradford and he has professional experience and qualifications in accounting and finance. He is currently the Chair Professor of Finance at Lingnan University.

About the Author—ERIC C.C. TSANG received the B.Sc. degree in Computer Studies from the City University of Hong Kong in 1990 and Ph.D. degree in computing at the Hong Kong Polytechnic University in 1996. He is an Assistant Professor of the Department of Computing of the Hong Kong Polytechnic University.

About the Author—XI-ZHAO WANG received the Ph.D. degree in Computer Science from Harbin Institute of Technology, Harbin, China, in 1998. Since 2001, he has been the Dean and Professor of the Faculty of Mathematics and Computer Science, Hebei University, China.