Neurocomputing 71 (2008) 3114–3123

Contents lists available at ScienceDirect

Neurocomputing

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

Enhanced feature selection models using gradient-based and point injection techniques

D. Huang, Zhaohui Gan, Tommy W.S. Chow *

Department of Electrical Engineering, City University of Hong Kong, Hong Kong

ARTICLE INFO

Available online 20 June 2008

Keywords:
Filter feature selection model
Gradient-based learning
Genetic algorithms
Point injection
Sequential forward search

ABSTRACT

This paper focuses on enhancing the effectiveness of filter feature selection models from two aspects. First, feature-searching engine is modified based on optimization theory. Second, a point injection strategy is designed to improve the regularization capability of feature selection. The second topic is important, because overfitting is usually experienced. To evaluate the proposed strategies, we implement these strategies to modify two classic filter feature selection models. One model is based on sequential forward search scheme and the other employs genetic algorithms (GA) for feature selection. Comparing the original and modified models on synthetic and real data, the contributions of our modification are shown.

1. Introduction

Huge amount of data are accumulated in an enormous speed unprecedentedly experienced in human history because computer technology became so advanced making the storage of large dataset possible. In some advanced engineering and physical science applications, most conventional computational methods have already experienced difficulty in handling the enormous data size. Feature selection is an essential and widely used technique to deal with the large data size problem. It reduces the number of features through eliminating irrelevant and redundant features, and thus results in increased accuracy, enhanced efficiency, and improved scalability for classification and other applications such as data mining [15]. Feature selection is especially important when one is handling a huge dataset with dimensions up to thousands.

A feature selection framework generally consists of two parts: a searching engine used to determine the promising feature subset candidates, and a criterion used to determine the best candidate [23,25]. There are several searching engines: ranking, optimal searching, heuristic searching, and stochastic searching. Among these engines, heuristic searching, which can easily be implemented and can deliver respectable results [29] is widely used. Feature selection models can be broadly categorized as filter model, wrapper model, and embedded model according to their evaluation criteria. To check the quality of features, filter models explore various types of statistical information, such as distribution probabilities underlying data, while wrapper and embedded models depend on the results of a specific classifier. To evaluate a feature subset, say S, wrapper and embedded models firstly require to build a classifier based on S. Wrapper models then rely on the performance of the built classifier to determine the goodness of S, while embedded models make use of the parameters of the built classifier to assess S. Wrapper/embedded models are usually more computationally expensive than filter models.

In filter model, good feature selection results rely on a respectable evaluation criterion and an appropriate searching strategy. The former issue has been heavily investigated. Various types of information, including mutual information [3,5,7], correlation [14], etc., have been explored for evaluating features. In contrast, much less attention has been paid to search engines. Also, most of the search-engine studies are conducted in discrete feature domains. For example, sequential forward searching (SFS), a typical heuristic searching scheme, identifies k important features from unselected features and places them into a selected feature subset in each iteration. To improve SFS, a stepwise strategy is designed—in each iteration, selecting k "good" unselected features is followed by deleting r "worst" selected features ($r<k$) [29]. Al-Ani and Deriche [1] employ "elite" selected features, not all of them, to identify important features from unselected ones. These algorithms, studied in a discrete feature space, depend on the testing of more feature combinations in order to deliver improved results. But it is clear that more testing will increase the computational complexity accordingly. Besides heuristic schemes, stochastic algorithms, such as genetic algorithms (GA), are also commonly used for feature search [10]. In the early work of this area, Siedlecki and Sklansky demonstrated that GA outperformed typical heuristic algorithms [31] in search

* Corresponding author. Tel.: +852 2788 7756; fax: +852 2788 7791.
E-mail address: eetchow@cityu.edu.hk (T.W.S. Chow).

0925-2312/$ - see front matter © 2008 Published by Elsevier B.V.
doi:10.1016/j.neucom.2008.04.023

effectiveness. Many subsequent works showing the advantages of GA in feature selection were reported [6,21,30,35]. Using the crossover and mutation operators, GA can escape from local optima, and can explore a wide range of search space when the parameters are properly controlled. However, they are weak in fine-tuning when the search is near local optimum points, which results in a relatively large running time. In order to improve the local search capability, local search methods should be considered to hybridize with simple GA. Studies about hybrid GA for feature selection are then emerging [26].

Given a set of n samples $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, which is drawn from a joint distribution P on $X \times Y$, a feature selection process is per se a learning process in the domain of $X \times Y$ to optimize the employed feature evaluation criterion, say $L_{(x,y)\sim P}(x, y)$. As P is unknown, $L_{(x,y)\sim P}(x, y)$ has to be substituted by $L_{(x,y)\in D}(x, y)$. Clearly, when D cannot correctly represent P, this substitution may cause overfitting in which the selected features are unable to deal with testing data satisfactorily despite performing splendidly on the training data D [4]. In many applications, a machine-learning process suffers from insufficient learning samples. For instance, in most microarray gene expression datasets, 10 samples are given. With such a small sample set, overfitting is likely to happen. Thus, the issue of overfitting must be addressed. A wrapper/embedded feature selection model always involves with classification-learning processes. The regularization techniques developed for classification learning can be directly employed in a wrapper/embedded model. For example, support vector machine and penalized Cox regression model, which have been argued to have high generalization capability, are employed in embedded models [12,13]. And an embedded feature selection model is trained based upon with regularized classification loss functions [28]. On the other hand, as filter schemes do not explicitly include a classification-learning process, the regularization techniques developed for classification learning cannot be explored. Thus, it is needed to design specified regularization strategies. To our knowledge, research on this topic is handful. In order to address the problem of overfitting, a bootstrap framework has been adopted for mutual information estimation [37]. Under such a framework, mutual information estimation should be conducted several times in order to deliver the final result. Apparently, the bootstrap framework is highly computationally demanding that precludes it from being widely used.

In this paper, we propose two strategies—the first one is aimed at improving the effectiveness of searching engines, and the second strategy is developed for addressing the overfitting issue. We choose two typical filter feature selection models as examples to demonstrate these strategies. These models employ Bayesian discriminant (BD) criterion [18,19] for feature evaluation, and use SFS [9] and GA as search engine. We firstly analyze the classical search engines, i.e., SFS and GA, according to the well-established optimization theory [4]. This type of analysis, which has been overlooked in previous studies, reveals the shortcoming of conventional SFS and the simple GA—they are unable to perform optimization in a maximal way. To address this issue, we naturally come to the optimization theory for solution. As a result, we propose a strategy to modify SFS and GA. With this strategy, feature search can be conducted along the more effective optimization direction. To enhance the regularization capability, a point injection approach is employed. This approach generates certain points according to the distribution of given samples. This is similar to the ones developed for classification learning. The injected points are employed for evaluating the feature subsets. This mechanism is able to minimize the undesired side effect of injected points.

In the next section, the Bayesian discriminant feature evaluation criterion and two popular types of search engine, sequential forward search and GA, are briefed. After that, our proposed strategies are described. Finally the proposed strategies are extensively evaluated.

2. Bayesian discriminant-based feature search

2.1. Bayesian discriminant feature evaluation

Assume that the feature set of n-sample dataset D is $F = \{f_1, f_2, \ldots, f_M\}$. Also, each pattern (say, x_i) falls into one of the L categories, i.e., $y_i = \omega_k$, where $1 \leqslant i \leqslant n$ and $1 \leqslant k \leqslant L$.

In filter models, probability-based feature evaluation criteria are commonly used. BD, a typical probability-based approach, is developed by Huang and Chow [18]. With the dataset D, BD is defined as

$$BD(S) = \frac{1}{n}\sum_{i=1}^{n} \log \frac{p_S(y_i|x_i)}{p_S(\overline{y}_i|x_i)} = \frac{1}{n}\sum_{i=1}^{n} \log \frac{p_S(y_i|x_i)}{1 - p_S(y_i|x_i)}, \quad (1)$$

where $\overline{y}_i$ means all the classes but class y_i, and $p_S(\cdot)$ represents a probability which is estimated by using the feature subset S. As shown in Eq. (1), $BD(S)$ directly measures the likelihood of given samples being correctly recognized by using S. A large $BD(S)$, which indicates that most given samples can be correctly classified, is preferred.

And in our study, the probabilities required by $BD(S)$ are estimated with Parzen window [27], which is modeled as

$$p(x, y) = \sum_{\text{all}(x_i,y_i)\in \text{class } y} p(x_i)p(x|x_i) = \sum_{y_i = y} p(x_i)\kappa(x - x_i, h_i), \quad (2)$$

$$p(x) = \sum_{\text{all classes}} p(x, y) = \sum_{\text{all}(x_i,y_i)\in D} p(x_i)\kappa(x - x_i, h_i), \quad (3)$$

where κ and h_i are the kernel function and the width of window, respectively. The Parzen window estimator (2) or (3) has been shown to be able to converge the real probability when κ and h_i are selected properly [27]. κ is required to be a finite-value nonnegative function and satisfies $\int \kappa(x - x_i, h_i)\,dx = 1$. And the width of κ, i.e., h_i, is required to have $\lim_{n\to\infty} h = 0$, where n is the number of given samples. Following the common way, we choose Gaussian function as κ. That is,

$$\kappa(x - x_i, h_i) = G(x - x_i, h_i) = \frac{1}{(2\pi h_i^2)^{M/2}} \exp\left(-\frac{1}{2h_i^2}(x - x_i)(x - x_i)^{\mathrm{T}}\right),$$

where M is the dimension of x. And the window width h_i is set with $h_i = 2\text{distance}(x_i, x_j)$, where x_j is the third nearest neighbor of x_i. We use Euclidean distance, i.e., $\text{distance}(x_i, x_j) = \sqrt{(x_i - x_j)(x_i - x_j)^{\mathrm{T}}}$ for two data vectors x_i and x_j. As to $p(x_i)$ of the Eqs. (2) and (3), it is estimated with $p(x_i) = 1/n$. With the Eqs. (2) and (3) and based on $p(y|x) = p(x, y)/p(x)$, $p(y|x)$ required by $BD(S)$ is finally obtained.

2.2. Feature search engines

A BD-based feature selection process is aimed at determining the feature subset S that can maximize $BD(S)$ (Eq. (1)). In general, $BD(S)$ is optimized in the following way: after a pool of feature subsets is suggested by a searching engine, BD of each suggested feature subset is calculated, and one with the largest BD is either outputted as the final feature selection result or remembered as the reference to guide the subsequent feature selection process. Many schemes for determining feature subset

pools have been developed to trade the quality of optimization results with computational consumption. Among them, heuristic search and stochastic search are popular. Sequential forward search is a typical heurist search engine, while GA is a classic stochastic search scheme. In this paper, these two are investigated.

2.2.1. Sequential forward search

First, we set the selected feature set (denoted by S, below) empty and enrich S through adding k important features iteratively into S. In each iteration, k features (generally, $k=1$) are selected in a way that all the feature combinations {S, k unselected features} are examined. The one with the largest BD is then extracted and updated as a new S. Based upon the S, another iteration of feature selection is conducted similarly. The process continues until certain stopping criteria are met. Given a feature set F for selection, SFS can be summarized as following:

- *Step* 1: Set the selected feature subset S with empty.
- *Step* 2: Repeat the following until certain stopping conditions are met.
  Identify the most useful feature (say, f_u) from the unselected genes and place it into S. fu satisfies $f_u = \arg\max_{f\in\{F-S\}} BD(S+g)$.

2.2.2. Genetic algorithms

GA is a biologically inspired approach simulating natural evolution [17]. It has been widely applied in numerous scientific and engineering optimization problems or search engines. The GA includes the following steps.

- *Step* 1: *Chromosome encoding*: In feature selection problems, we use a variable length integer string as chromosome encoding. An integer in integer string represents a feature. A value of an integer represents the number of a feature. For example, chromosome 3267, 654, 2109 means that the 3267th, 654th, and 2109th features are selected, and all the other features are removed. The length of a chromosome is the number of features selected, which is determined by users.
- *Step* 2: *Fitness evaluation*: The objective of feature selection is to optimize an evaluation criterion. In this study, we use BD [18] as the evaluation criterion. The fitness of a chromosome S (i.e., a feature subset) is defined as *fitness* $(S) = BD(S)$.
- *Step* 3: *Selection and stop*: The chromosome selection for the next generation is conducted on the basis of fitness. The selection mechanism should ensure that fitter chromosomes have a higher probability of survival. In our design, we use the rank-based selection scheme. When the number of running generations reaches the preset value, the GA stop.
- *Step* 4: *Crossover and mutation*: The standard single point crossover and mutation operators are used. They choose one cutting point randomly and alternately copy each segment out of two parents.

3. Modified strategies

3.1. Weighting sample

The objective of feature selection is to optimize the employed evaluation criterion, for example, $BD(S)$ (Eq. (1)) in this study, through adjusting S. To clearly explain our idea, we recast $BD(S)$ (1) as

$$BD(S) = \frac{1}{n}\sum_{i=1}^{n} \log \underbrace{\frac{p_S(y_i|x_i)}{1-p_S(y_i|x_i)}}_{f((x_i,y_j),S)} = \frac{1}{n}\sum_{i=1}^{n} \log f((x_i,y_i),S). \tag{4}$$

According to the optimization theory, the steepest direction of adjusting S to maximize Eq. (4) is determined by

$$\frac{\partial BD(S)}{\partial S} = \frac{1}{n}\sum_{i=1}^{n} \frac{\partial BD(S)}{\partial f((x,y),S)} \frac{\partial f((x,y),S)}{\partial S}\bigg|_{(x_i,y_i)}. \tag{5}$$

It shows that, to optimize $BD(S)$, the updating of S depends on the two terms, $\partial BD(S)/\partial f((x,y),S)$ and $\partial f((x,y),S)/\partial S$. The former one happens in a continuous domain, while the latter one is related to S and has to be tackled in a discrete feature domain. In this sense, Eq. (5) cannot be solved directly. To maximize $BD(S)$, SFS tests all combinations of S and unselected features, and selects the one having the maximal BD. Clearly, SFS only considers the second term of Eq. (5), but overlooks the first term. It means that the searching direction of SFS is not according to the steepest optimization one. The effectiveness of optimization will then be degraded as a result.

Naturally, our proposed strategy is based on the optimization theory, i.e., Eq. (5). The second term of Eq. (5) is resolved by using any conventional discrete-domain searching scheme. We use SFS for this purpose. The first term of Eq. (5) can be directly calculated in the way of

$$\frac{\partial BD(S)}{\partial f((x,y),S)} = \frac{\partial \log(f(x,y),S)}{\partial f((x,y),S)} = \frac{1}{f((x,y),S)} = \frac{1-p_S(y|x)}{p_S(y|x)}. \tag{6}$$

This shows that $\partial BD(S)/\partial f((x,y),S)$, which is only related to x, is independent of the change making on S. With this observation, we use Eq. (6) as weights to samples. In such way, feature searching is conducted on the weighted samples, but not on the original ones.

Assume that the dataset D is weighted by $\{w_1, w_2, \ldots, w_n\}$. With this weighted dataset, the criterion BD (Eq. (1)) and the probability estimations (2) and (3) are adjusted accordingly. The rule of $p(x_i) = 1/n$ is replaced by $p(x_i) = w_i/n$. Also, we have

$$p(x,y) = \sum_{\text{all}(x_i,y_i)\in\text{class } y} \frac{w_i}{n} G(x-x_i, h_i). \tag{7}$$

And the criterion BD is modified as

$$\begin{aligned} BD(S) &= \frac{1}{n}\sum_{i=1}^{n} w_i \log \frac{p_S(y_i|x_i)}{1-p_S(y_i|x_i)} \\ &= \frac{1}{n}\sum_{i=1}^{n} w_i \log \frac{p_S(x_i,y_i)}{\sum_{\text{all } y_j\neq y_i} p_S(x_i,y_j)}. \end{aligned} \tag{8}$$

Apparently, it is natural to regard that different samples may have different contributions to the learning processes. It is worth noting that most current machine-learning algorithms have already incorporated this idea. For instance, the classification learning is aimed at minimizing the mean squared error $L(\Lambda) = \sum_{\text{all}(x_i,y_i)}(f(x_i,\Lambda)-y_i)^2$ by training the model f, i.e., adjusting the parameter set Λ of f. The steepest decent type algorithm, commonly used for classification learning, determines the updating direction with

$$\begin{aligned} -\frac{\partial E}{\partial \Lambda} &= \sum_{\text{all}(x_i,y_i)} -\frac{\partial E}{\partial f}\frac{\partial f(x,\Lambda)}{\partial \Lambda}\bigg|_{x=x_i,y=y_i} \\ &= \sum_{\text{all}(x_i,y_i)} -(f(x,\Lambda)-y)\frac{\partial f(x,\Lambda)}{\partial \Lambda}\bigg|_{x=x_i,y=y_i}, \end{aligned} \tag{9}$$

where (x_i,y_i) is a given training sample. It is noted that the contribution of (x_i,y_i) is penalized by $|f(x_i,\Lambda)-y_i|$. AdaBoosting [16] is another example of a typical boosting learning algorithm that

weights the sample (x_i, y_i) with $w_i e^{-y_i f(x_i)}$ repeatedly during the course of learning, where w_i is the current weight to (x_i, y_i). Also, in order to reduce the risk of overfitting, it is intuitively expected that the negative samples (i.e., incorrectly recognized ones) have more influence to the subsequent learning than the positive samples. As a result, the convergence rate can be speeded up, and the problem of overfitting can also be alleviated [22]. It is clear that using AdaBoosting algorithm meets the above expectation. But Eq. (9) indicates that the steepest decent algorithm fell short on tackling overfitting in a way that the correctly recognized patterns still carry large weights. This shows the need on deriving improved versions of gradient-based algorithms [22]. Consider our proposed weighting-sample strategy as defined in Eq. (6). It penalizes the negative patterns heavily, which induces an effect on alleviating the problem of overfitting.

3.2. Point injection

Overfitting is caused by the deviation between the real optimization goal and the actual achievable optimization objective. The real goal of the BD-based feature selection process is to maximize $BD_P(S)$, where P is the underlying probability. As P is unknown in most cases, $BD_P(S)$ can not be actually defined, and thus has to be substituted with its empirical estimate $BD_D(S)$ (simplified as $BD(S)$, like Eq. (1) does). When $BD(S)$ cannot always reflect $BD_P(S)$ correctly, overfitting is caused. To avoid overfitting, it is preferred that $BD_P(S)$ varies smoothly enough.

In the area of classification/regression, the effect of overfitting can be alleviated through modifying the employed empirical objective function with regularization terms. These regularization terms penalized the complex models. Incorporating these regularization terms, relatively simple models can be obtained. Thus, the chance of suffering from overfitting will be reduced [4]. It must be pointed out that deriving the penalty terms has never been straightforward, because it requires thorough theoretical analysis. This is especially the case when the parameters or factors controlling smoothness of a training model are hard to determine. Another widely used regularization technique is point injection. It is known that smoothness means that samples close to each other should correspond to similar performance. This is the rationale behind the point injection technique. In many literatures, point injection technique is referred as *noise injection* [24,33,36], but the injected points are certainly not expected to be real noise. In order to avoid the confusion, we use the term *point injection* instead of *noise injection* in this paper.

Under the frameworks of classification/regression learning, injected points are always treated like the original samples—a classifier/regression model is built upon the original samples as well as the injected points. This working mechanism requires high-quality points. Spherical Gaussian distributed points are generated around each training object [4,24], but the added points may have increased the complexity. Thus, high-quality injected points, such as k-NN direction points [33] and eigenvector direction points [36], are suggested to replace Gaussian distributed points. Also, points are generated in a way of feature-knockout [34]. With improved quality of injected points, the advantages of injected point techniques are enhanced. In this study, the risk caused by point injection is reduced through adopting a different mechanism. Under our proposed mechanism, only the given samples are used for deriving the probability estimators. The given samples and the injected points are used for evaluating the feature subsets. Without participating in the process of model building, the undesirable effect of injected points is then reduced.

Around a pattern x_i, the point injection technique adds v points that are generated from a distribution $b(x-x_i)$. v and $b(x-x_i)$ play important parts in a point injection scheme [20,33]. In order to strike the balance between performance stability and computational efficiency, v is determined. Also, it has been argued that, for the reasonable choice of v, such as $v = 8$, 10 or 20, the effect of point injection is slightly different [33]. We thus set $v = 10$. As to $b(x-x_i)$, the "width" of $b(x-x_i)$, which determines the variance of the injected points, is crucial. As the aim of point injection is to test the properties of the region around x_i, too large a width of $b(x-x_i)$ is not desired, while too small a value of $b(x-x_i)$ may not have significant effect.

To determine an appropriate width of $b(x-x_i)$, the simulation-based strategies is used [33]. Inspired by the ideas described in Refs. [11,20], we develop an analytic approach for determining the width of $b(x-x_i)$. Aiming to reduce the bias intrinsic to the re-substitution error estimation as much as possible [11], our approach depends on the joint distribution (X, C) to determine the width of $b(x-x_i)$. Around a given pattern, say x_i, several points around it are generated from Gaussian distribution $N(x_i, \sigma_i)$, where $\sigma_i = d_i/2$ and d_i is the distance of x_i to the nearest samples, i.e.,

$$d_i = \arg\min_{j, j\neq i} ||x_i - x_j||. \tag{10}$$

In this way, it can assure that x' having $||x_i - x'|| = d_i$ occurs with the close-zero probability.

The given sample set D cannot cover each part of the whole data domain very well. In turn, the probability estimators built with these samples cannot describe every part of the data domain. In detail, there may exist the parts that the conditional probabilities $p(x|y)$ for all classes are small. According to Eq. (8), it can be shown that $|(\partial BD(S))/(\partial p(x,\omega))| \propto 1/(p(x,\omega))$ for all classes ω. It indicates that, when all $p(x|y)$ are small, a little change of x will cause a large change of $BD(S)$. The points of such type are not expected.

For the originally given samples on which the probability models are built, at least one $p(x|y)$ must be large enough. On the other hand, an injected point may be uncertain. That is, all the probabilities about it are very small. It is better to minimize the impact of uncertain points, although it can be argued that they may equally affect the quality of different feature subset candidates. Using this idea, the way of calculating $BD(S)$ of injected points is modified. Suppose that, according to the given D, we generate dataset D'

$$BD(S)|_{D'} = \frac{1}{|D'|} \sum_{\text{all}(x'_i, y'_i)\in D'} w'_i \log \frac{p_S(y'_i|x'_i)}{p_S(\bar{y'}_i|x'_i)} \underbrace{\left(\arg\max_{\text{all }\omega}(p_S(x'_i|\omega))\right)}_{\text{A}}, \tag{11}$$

where $|D'|$ means the cardinality of D'. y'_i and w'_i are the weight and class label of x'_i, respectively, and are inherited from the corresponding sample in D. With part A, the impact of uncertain points will be limited. This satisfies our expectation.

Below, contributions of the point injection strategy are assessed on a group of three-class and eight-feature synthetic datasets. In these data, the first four features are generated according to

- Class 1$\sim m$ samples from $N((1,1,-1,-1),\sigma)$,
- Class 2$\sim m$ samples from $N((-1,-1,1,1),\sigma)$,
- Class 3$\sim m$ samples from $N((1,-1,1,-1),\sigma)$.

And the other four features are randomly determined from normal distribution with zero means and unit variance. Clearly, among the total eight features, the first four are equally relevant to the classification task, and the others are irrelevant. Three feature selection methods are applied to this data to determine

four salience features. They are the conventional SFS, SFS with the feature-knock-out, and SFS with the proposed point injection approaches. Only if all relevant features are selected, the selection results can be considered correct.

Different settings of σ and m are investigated. For reliable estimation, in each setting, three feature selection methods are run on 10,000 datasets independently generated. And the correct results over 10,000 trials are counted. In Table 1, the correctness ratios are presented. It shows that the feature-knock-out point injection strategy cannot bring the improved feature selection results in this example. This may due to the fact that the strategy is originally designed for classification learning, not for feature selection. In our proposed point injection strategy, its advantage becomes more significant either when the sample size becomes small or when σ becomes large. All these conditions actually mean that there is a high possibility of overfitting, because the larger σ mean is, the more complex the problem is. Thus, the presented results can suggest that our proposed approach can improve the generalization capability of SFS.

3.3. Procedures

This section shows the implementation of the proposed strategies by modifying conventional search engines.

3.3.1. Modified SFS

With the above weighting sample and point injection strategies, the conventional BD-based SFS feature selection model is modified as follows.

- *Step* 1 (*Initialization*): Set the selected feature set S and injected point set D' empty. Also, for each sample, assign a weight of 1, i.e., $w_i = 1$, $1 \leqslant i \leqslant n$.
- *Step* 2 (*Feature selection*): From the feature set F, identify the feature f_m which satisfies

$$f_m = \arg\max_{f \in F} [BD(f+S)|_D + BD(f+S)|_{D'}].$$

  The probability estimators required by BD are established with Eq. (7) based on the dataset D. And the BDs on D and D' are defined in Eqs. (8) and (11), respectively. Put the feature f_m into S and delete it from F at the same time.
- *Step* 3 (*Update the sample weights*): Set w_i based on Eq. (6). Then normalize w_i as $w_i = w_i / \sum_{j=1}^{n} w_j$.
- *Step* 4 (*Point injection*): Set D' empty. In the data domain described by S, conduct point injection around each sample in the following way.
  Around the pattern x_i, produce 10 points based on the distribution $N(x_i, d_i/2)$, where d_i is defined by the Eq. (10). Place these points into D'. Also, the class label and the weight of these injected points are set with y_i and w_i, respectively.
- *Step* 5: If the size in S has reached the desired value, then stop the whole process and output S, otherwise go to step 2.

Table 1
Comparisons on a synthetic data

	SFS	SFS with feature-knock-out strategy	SFS with the point injection strategy
$\sigma = 0.3$			
$m = 3$	0.980	0.941	0.991
$m = 9$	1.000	1.000	1.000
$\sigma = 0.8$			
$m = 3$	0.357	0.358	0.392
$m = 9$	0.933	0.930	0.931

These results demonstrate the merits of the proposed point injection strategy.

3.3.2. Modified GA

In order to further examine the performance of the above-described weighting sample and point injection strategies, the conventional BD-based GA feature selection model is also modified. We add a steepest descent operator in the simple GA. With the weighting-sample strategy, our modified GA can search the local optimal value efficiently, which is stated as follows.

- *Step* 1 (*Initialization*): Set the selected feature set S empty. Randomly generate N the initial population. Also, for each sample, assign a weight of 1, i.e., $w_i = 1$, $1 \leqslant i \leqslant n$.
- *Step* 2 (*Point injection*): Set D' with empty. Around the pattern x_i, produce 10 points based on the distribution $N(x_i, d_i/2)$, where d_i is defined by the Eq. (10). Place these points into D'. Also, the class label and the weight of these injected points are set with y_i and w_i, respectively.
- *Step* 3 (*Simple GA for feature selection*):
  (a) *Selection*: the fitter chromosomes are selected from the N individuals in population for evolution. The fitness of a chromosome S is defined as $\text{fitness}(S) = BD(S)|_D + BD(S)|_{D'}$. The probability estimators required by BD are established with Eq. (7) based on the dataset D. And the BDs on D and D' are defined in Eqs. (8) and (11), respectively.
  (b) *Crossover*: choose one cutting points at random and alternately copies each segment out of the two parents.
  (c) *Mutation*: determine which gene of a chromosome in population will mutate, and change it as another gene.
- *Step* 4 (*Steepest descent operator*): Set w_i based on Eq. (6). Then normalize w_i as $w_i = w_i / \sum_{j=1}^{n} w_j$.
- *Step* 5: If the maximal generation has reached, then stop the whole process and output S, otherwise go to step 3.

4. Experimental results

In this study, the proposed strategies are used to modify SFS and GA. In order to elaborate the results, we first evaluate the modified SFS.

Below, the modified scheme SFS, called gradient and point injection-based SFS (gp-SFS), is evaluated by comparing with several related methods, namely, the conventional SFS, support machine learning recursive feature elimination scheme (SVM RFE) [13], and the conventional SFS with the feature-knock-out regularization technique (fko-SFS) [34]. SVM RFE, a typical embedded feature selection model, begins with the training of an SVM (of linear kernel) with all the given features. Then according to the parameters of the trained SVM, features are ranked in terms of importance, and half of the features are eliminated. The training-SVM-eliminating-half-of-features process repeats until no feature is left. As SVM RFE employs the classic SVM model to conduct feature selection, the SVM RFE method is not applicable to the multi-class vehicle database.

The feature-knock-out point injection scheme is designed for classification learning in which a point x' is added in each learning iteration. To generate x', two samples (say x_1 and x_2) are randomly selected and a feature f is specified according to the newly built model. And all information about x' is set with that of x_1, except that of $x'(f) = x_2(f)$. We adopt this point injection scheme to modify the conventional SFS as fko-SFS.

In this study, we rely on experimental classification results to assess the quality of feature selection results. In detail, given a feature subset for examining, say S, certain classifiers are constructed using training data, which is also used for feature selection. Then, based on the performance of these classifiers on a

test dataset, the quality of S is evaluated. Respectable feature subsets should correspond to good classification results. In our study, we apply the feature selection methods to four datasets, and use four classifiers to evaluate the feature selection results.

4.1. Datasets

4.1.1. Sonar classification

It consists of 208 samples. Each sample is described with 60 features and falls into one of two classes, metal or rock. From 208 samples, 40 ones are randomly selected for training and the others are used for test.

4.1.2. Vehicle classification

This is a four-class dataset for distinguishing the type of vehicle. There are totally 846 samples provided. Each sample is described with 18 features. We randomly select 80 samples for training. The remaining 766 samples are used for testing. It must be noted that because of the nature of the basic SVM, the basic SVM classification models cannot be applied to this multi-class dataset.

4.1.3. Colon tumor classification

This is a microarray dataset and is built for colon tumor classification, which contains 62 samples collected from colon-cancer

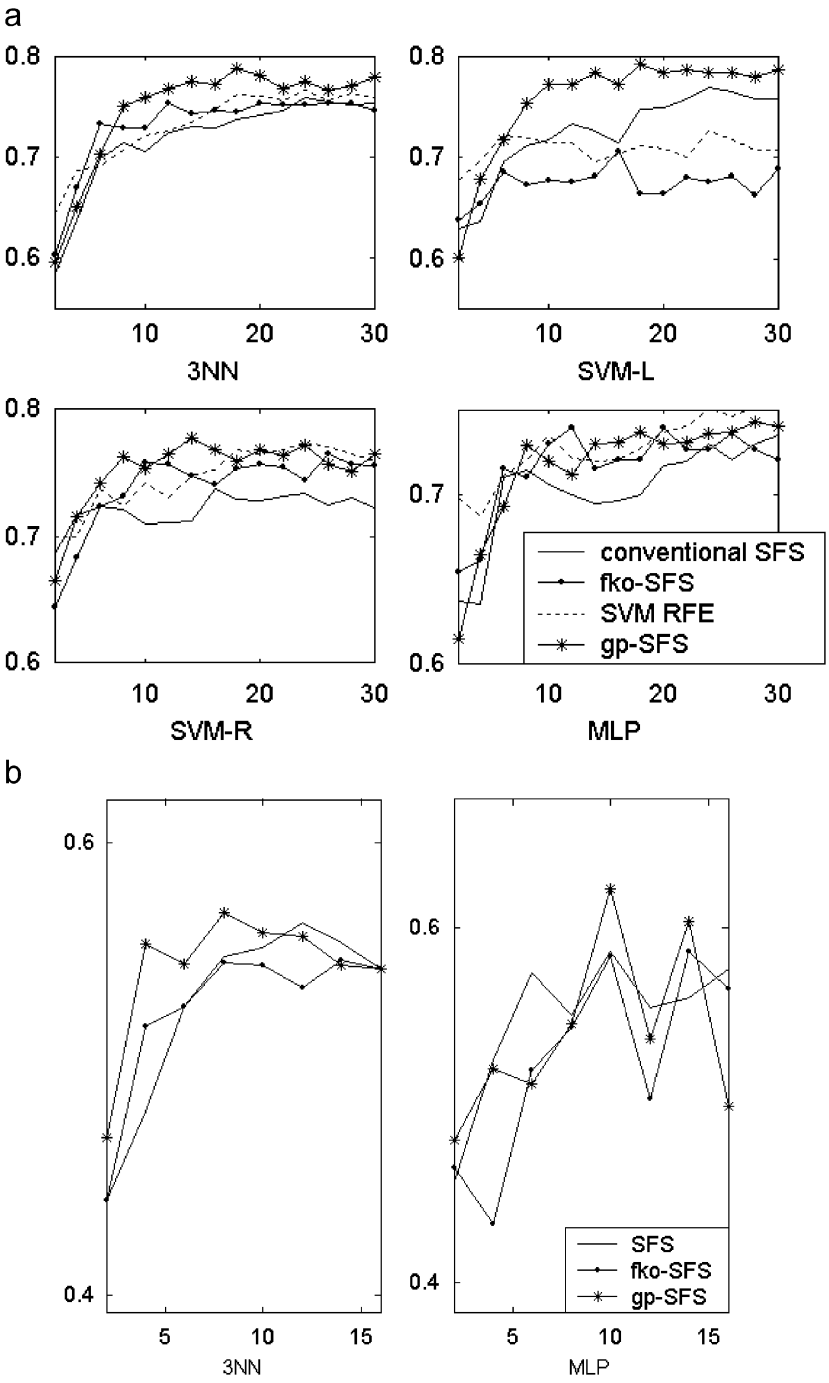


Fig. 1. Comparison results on UCI datasets under SFSs framework against number of features. (a) Classification accuracy rate of four classifiers against number of selected feature using four different feature selection methods for sonar dataset. (b) Classification accuracy rate of two classifiers against number of selected feature using three different feature selection methods for vehicle classification.

patients [2]. Among these samples, 40 samples are tumor, and 22 are labeled "normal". There are 2000 genes (features) selected based on the confidence in the measured expression levels. We randomly split the 62 samples into two disjoint groups—one group with 31 samples for training and the other one with 31 samples for test.

4.1.4. Prostate cancer classification

This is another microarray dataset, which is collected with the aim to identify prostate cancer cases from noncancer cases [32]. This dataset consists of 102 samples from the same experimental conditions. And each sample is described by using 12,600 genes (features). We split the 102 samples into two disjoint groups—one group with 60 samples for training and the other with 42 samples for testing.

4.2. Classifiers

Four typical classifiers employed in this study include multilayer perceptron model (MLP), support vector machine model with linear kernel (SVM-L), the support vector machine model with RBF kernel (SVM-R), and the 3-NN rule classifier. The MLP used in our study is available at: 〈http://www.ncrg.aston.ac.uk/netlab/〉. For convenience, we set six hidden neurons of MLP for all examples. It must be noted that slightly different numbers of hidden neuron do not have an effect on the overall performance. The number of training cycles is set with 100 in order to ease the concerns on overfitting. And other learning parameters are set with default values. SVM models are available at: 〈http://www.isis.ecs.soton.ac.uk/resources/svminfo〉. The default setting of C for SVM is used. That is, C is set as Inf. We set the width of RFB kernel to 1 with consideration that all the data are normalized to unit variance and zero mean before classification. In this study, we use the widely used SVM models, which can handle only two-class classification problems [8]. Thus, there is no SVM classification result for the case of vehicle dataset.

4.3. Evaluations of the modified SFS

In each example, we repeat investigation on 10 different sets of training and test data. The presented results are the statistics of 10 different trials. Also, in each training data, the original ratios between different classes are roughly remained. For example, during the investigation on the colon cancer classification, the original ratio between tumor and normal class, i.e., 40 normal vs. 22 tumor, is roughly kept in each training dataset. For each training dataset, we preprocess it so that each input variable has zero means and unit variance. And the same transformation is then applied to the corresponding test dataset.

The computational complexity of SFS type models is $O(M^2)$, where M is the number of features. A microarray dataset generally contains information of thousands or ten thousands genes. Clearly, directly handling the huge gene sets cost SFSs unbearable computational burden. To improve the computational efficiency, and given by the fact that most genes originally given in a microarray dataset are irrelevant to a specified task, a widely used pre-filtering-gene strategy is adopted in our study to eliminate the irrelevant and insignificantly relevant genes before the commencement of feature selection. In detail, all the given features (genes) are ranked in a descending order of BD (Eq. (8)). And the one-third top-ranked features are left behind for further feature selection.

The comparative results are presented in Fig. 1 (for sonar classification and for vehiecle classification), Fig. 2 (for colon cancer classification), and Fig. 3 (for prostate cancer classification). In these figures, the x-axes give the number of selected features and y-axes indicate the classification accuracy rate of the trained classifiers on testing data. In most cases, our modified SFS greatly outperform the conventional SFS. This is contributed by the gradient-based method and point injection strategies. Also, compared with fko-SFS and SVM RFE in which the problem of small sample is tackled explicitly or implicitly, the proposed SFS still shows its advantages. This confirms the contributions of our proposed methods.

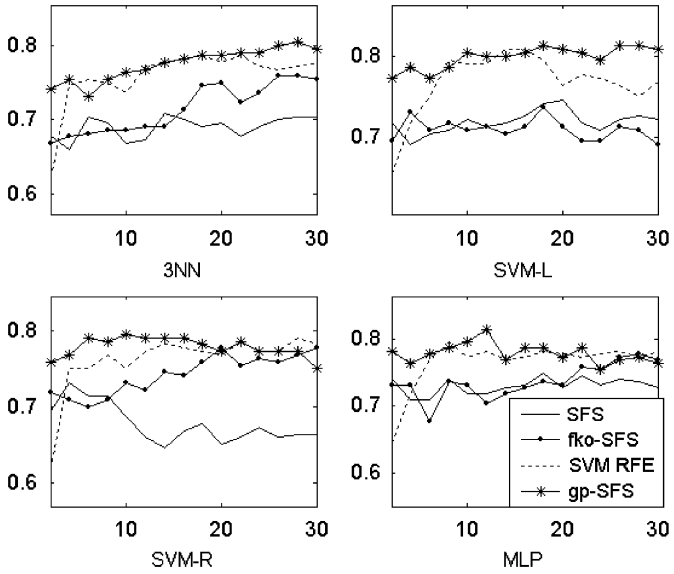


Fig. 2. Classification accuracy rate of four classifiers against number of selected feature using four different feature selection methods for colon cancer classification dataset under framework of SFSs.

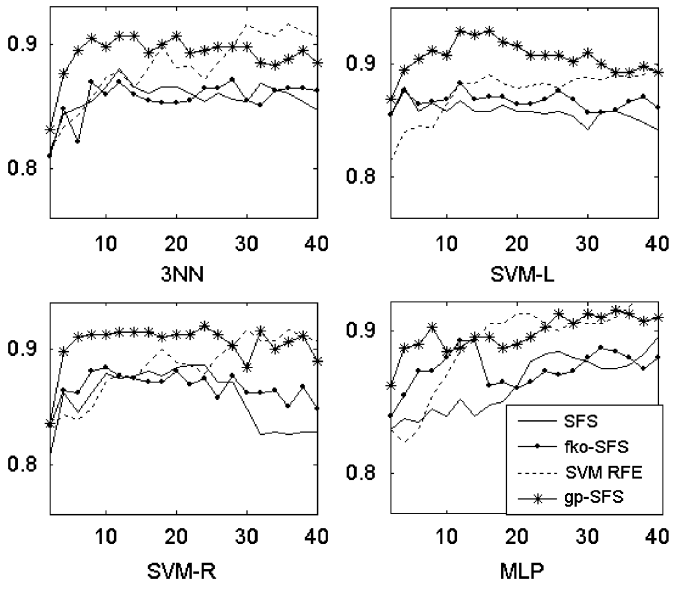


Fig. 3. Classification accuracy rate of four classifiers against number of selected feature using four different feature selection methods for prostate cancer classification dataset under framework of SFSs.

4.4. Evaluations of the modified GA

To evaluate the proposed strategies under the framework of GA, we compare the modified GA, called hybrid GA and point

injection (p-HGA) with the conventional SGA. For this purpose, the above classification evaluation schemes are used. That is, p-HGA and SGA are compared on the aforementioned datasets, and are evaluated by using the classifiers MLP, SVM, and kNN ($k = 3$). Also, we use SVM RFE as an evaluation baseline.

The comparative results are presented in Fig. 4 (for sonar classification and for vehiecle classification), Fig. 5 (for colon cancer classification), and Fig. 6 (for prostate cancer classification). In these figures, the x-axes give the number of selected features, and y-axes indicate the classification accuracy rate of trained classifiers on testing data. In most cases, the modified GA, i.e., p-HGA, outperforms the conventional GA. These confirm the contributions of the proposed strategies. For instance, in the example of colon cancer classification, apart from one case, the results of p-HGA are better than those of SGA in all other cases. It is also noticed that p-HGA has a higher accuracy rate of about 20%. Similar remarks can be obtained on other datasets.

5. Conclusions

In this paper, two strategies are proposed to enhance the performance of filter feature selection models. First, a gradient-

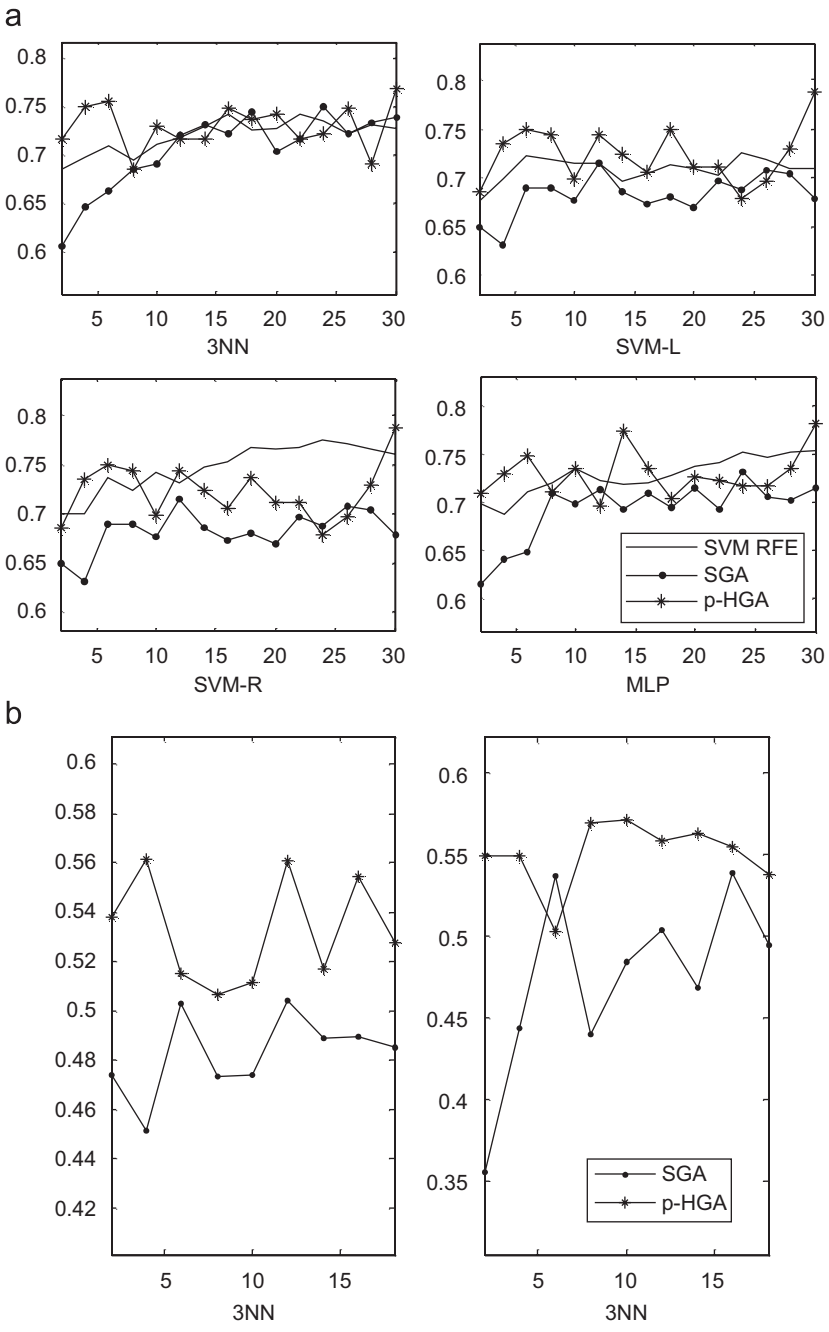


Fig. 4. Comparisons results on UCI datasets under GA framework. (a) Classification accuracy rate of four classifiers against number of selected feature using three different feature selection methods for sonar dataset. (b) Classification accuracy rate of two classifiers against number of selected feature using two different feature selection methods for vehicle classification.

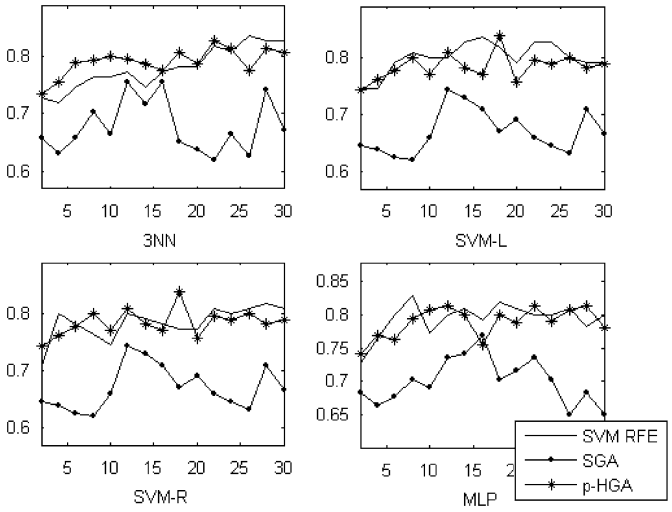


Fig. 5. Classification accuracy rate of four classifiers against number of selected feature using three different feature selection methods for colon cancer classification dataset under framework of GA.

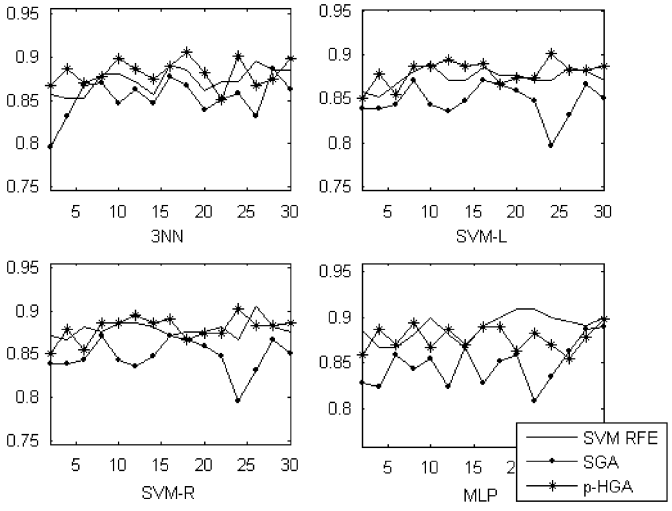


Fig. 6. Classification accuracy rate of four classifiers against number of selected feature using three different feature selection methods for prostate cancer classification dataset under framework of GA.

based strategy is used to enhance the searching effectivenss, and a new point injection approach is introduced to improve generalization ability. We apply these strategies to modify two typical filter models—SFS-based and GA-based approaches. We evalaute the modified models on synthetic data and real data. The obtained results demonstrate that these proposed strategies can bring improvement to either the SFS-based model or the GA-based model. It is worth noting that the improvement is quite remarkable in some cases. In furture work, we will further extend these strategies to other feature evaluation criteria and evaluate their merits and limitations.

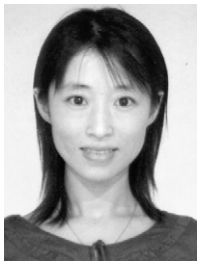
Acknowledgment

The work described in this paper was fully supported by a SRG grant from the City University of Hong Kong of Project no. 7001998-570.

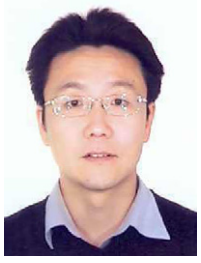
References

[1] A. Al-Ani, M. Deriche, Optimal feature selection using information maximisation: case of biomedical data, in: Proceedings of the 2000 IEEE Signal Processing Society Workshop, vol. 2, 2000, pp. 841–850.
[2] U. Alon, N. Barkai, D.A. Notterman, et al., Broad pattern of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays, Proc. Natl. Acad. Sci. USA 96 (12) (1999) 6745–6750.
[3] R. Battiti, Using mutual information for selecting features in supervised neural net learning, IEEE Trans. Neural Networks 5 (1994) 537–550.
[4] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, New York, 1995.
[5] B. Bonnlander, Nonparametric selection of input variables for connectionist learning, Ph.D. Thesis, University of Colorado at Boulder, 1996.
[6] F.Z. Brill, D.E. Brown, W.N. Martin, Fast genetic selection of features for neural network classifiers, IEEE Trans. Neural Networks 3 (2) (1992) 324–328.
[7] T.W.S. Chow, D. Huang, Estimating optimal feature subsets using efficient estimation of high-dimensional mutual information, IEEE Trans. Neural Networks 16 (1) (2005) 213–224.
[8] C. Cortes, V. Vapnik, Support vector machine, Machine Learning 20 (3) (1995) 273–297.
[9] P.A. Devijver, J. Kittler, Pattern Recognition: A Statistical Approach, Prentice-Hall, Englewood Cliffs, 1982.
[10] Z.H. Gan, T.W.S. Chow, D. Huang, Effective gene selection method using Bayesian discriminant based criterion and genetic algorithms, J. Signal Process. Syst. 50 (3) (2008) 293–304.
[11] N. Glick, Additive estimators for probabilities of correct classification, Pattern Recognit. 18 (2) (1985) 151–159.
[12] J. Gui, H. Li, Penalized Cox regression analysis in the high-dimensional and low-sample size settings, with application to microarray gene expression data, Bioinformatics 21 (13) (2005) 3001–3008.
[13] I. Guyon, J. Weston, S. Barnhill, Gene selection for cancer classification using support vector machines, Machine Learning 46 (2002) 389–422.
[14] M.A. Hall, Correlation-based feature selection for machine learning, Ph.D. Thesis, Department of Computer Science, Waikato University, New Zealand, 1999.
[15] J.W. Han, M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann Publishers, San Francisco, 2001.
[16] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, Springer, 2001, pp. 308–312.
[17] J. Holland, Adaptation in Nature and Artificial Systems, MIT Press, 1992.
[18] D. Huang, T.W.S. Chow, Efficiently searching the important input variables using Bayesian discriminant, IEEE Trans. Circuits Syst. 52 (4) (2005) 785–793.
[19] D. Huang, T.W.S. Chow, et al., Efficient selection of salient features from microarray gene expression data for cancer diagnosis, IEEE Trans. Circuits Syst. Part I 52 (9) (2005) 1909–1918.
[20] S. Kim, E.R. Dougherty, J.Y. Barrera, et al., Strong feature sets from small samples, J. Comput. Biol. 9 (2002) 127–146.
[21] L.I. Kuncheva, L.C. Jain, Nearest neighbor classifier: simultaneous editing and feature selection, Pattern Recognit. Lett. 20 (1999) 1149–1156.
[22] F. Lampariello, M. Sciandrone, Efficient training of RBF neural networks for pattern recognition, IEEE Trans. Neural Networks 12 (5) (2001) 1235–1242.
[23] H. Liu, H. Motoda, Feature Selection for Knowledge Discovery and Data Mining, Kluwer Academic Publishers, London, UK, 1998.
[24] S. Matsuoka, Noise injection into inputs in back-propagation learning, IEEE Trans. Syst. Man Cybern. 22 (1992) 436–440.
[25] L.C. Molina, L. Belanche, A. Nebot, Feature selection algorithms: a survey and experimental evaluation, Technical Report, 2002. Available from: 〈http://www.lsi.upc.es/dept/techreps/html/R02-62.html〉.
[26] I. Oh, J. Lee, B. Moon, Hybrid genetic algorithms for feature selection, IEEE Trans. Pattern Anal. Machine Intell. 26 (11) (2004) 1424–1437.
[27] E. Parzen, On the estimation of a probability density function and mode, Ann. Math. Stat. 33 (1962) 1064–1076.
[28] S. Perkins, K. Lacker, J. Theiler, Grafting: fast, incremental feature selection by gradient descent in function space, J. Machine Learning Res. 3 (2003) 1333–1356.
[29] P. Pudil, J. Novovicova, J. Kittler, Floating search methods in feature selection, Pattern Recognit. Lett. 15 (1994) 1119–1125.
[30] M.L. Raymer, W.F. Punch, E.D. Goodman, L.A. Kuhn, A.K. Jain, Dimensionality reduction using genetic algorithms, IEEE Trans. Evol. Comput. 4 (2) (2000) 164–171.
[31] W. Siedlecki, J. Sklansky, A note on genetic algorithms for large-scale feature selection, Pattern Recognit. Lett. 10 (1989) 335–347.
[32] D. Singh, et al., Gene expression correlates of clinical prostate cancer behavior, Cancer Cell 1 (2002) 203–209.
[33] M. Skurichina, S. Raudys, R.P. Duin, K-nearest neighbours directed noise injection in multilayer perceptron training, IEEE Trans. Neural Networks 11 (2) (2000) 504–511.
[34] L. Wolf, I. Martin, Regularization through feature knock out, AI memo 2004–2005. Available from: 〈http://cbcl.mit.edu/cbcl/publications/ai-publications/2004/〉, 2004.
[35] J.H. Yang, V. Honavar, Feature subset selection using a genetic algorithm, IEEE Intell. Syst. 13 (2) (1998) 44–49.
[36] N.G. Zagoruiko, V.N. Elkina, V.S. Temirkaev, ZET—an algorithm of filling gaps in experimental data tables, Comput. Syst. 67 (1976) 3–28.

[37] X. Zhou, X. Wang, E. Dougherty, Nonlinear probit gene classification using mutual information and wavelet-based feature selection, J. Biol. Syst. 12 (3) (2004) 371–386.



D. Huang received the Ph.D. degree in 2005 from the Department of Electronic Engineering of City University of Hong Kong. Her research focuses on machine learning and bioinformatics.



Zhaohui Gan obtained B.Eng. degree and M.Phil. degree in electrical engineering at Wuhan University of Science and Technology in 1990 and 1995, respectively. He is now working towards the Doctor of Philosophy at City University of Hong Kong. His research interest is evolutionary computation, pattern recognition, and applications.

Tommy W.S. Chow received the B.Sc. (1st Hons) degree and the Ph.D. degree from the Department of Electrical and Electronic Engineering, University of Sunderland, UK, in 1984 and 1988, respectively. He undertook his Trainee with Reyrolle Technology at Tyne and Wear, UK. His PhD research worked on a collaborative project between The International Research and Development, Newcastle Upon Tyne, UK, and the Ministry of Defense (Navy) UK. He is a professor in the Department of Electronic Engineering at the City University of Hong Kong, Hong Kong. He has been working on different consultancy projects with the Mass Transit Railway, Kowloon-Canton Railway Corporation, Hong Kong. He has also conducted other collaborative projects with the Kong Electric Co. Ltd, and Royal Observatory Hong Kong, and the MTR Hong Kong on the application of neural networks for machine fault detection and forecasting. He an author and co-author of numerous published works, including book, book chapters, and over 110 journal articles related to his research. His main research has been in the area of machine learning, optimizations, bioinformatics, and fault diagnostics. Prof. Chow received the Best Paper Award in 2002 IEEE Industrial Electronics Society Annual meeting in Seville, Spain. He was the Chairman of Hong Kong Institute of Engineers, Control Automation and Instrumentation Division 1997–1998. T.W.S. Chow IEEE Membership no. 03747557.