Correspondence

Wrapper–Filter Feature Selection Algorithm Using a Memetic Framework

Zexuan Zhu, Yew-Soon Ong, and Manoranjan Dash

***Abstract*—This correspondence presents a novel hybrid wrapper and filter feature selection algorithm for a classification problem using a memetic framework. It incorporates a filter ranking method in the traditional genetic algorithm to improve classification performance and accelerate the search in identifying the core feature subsets. Particularly, the method adds or deletes a feature from a candidate feature subset based on the univariate feature ranking information. This empirical study on commonly used data sets from the University of California, Irvine repository and microarray data sets shows that the proposed method outperforms existing methods in terms of classification accuracy, number of selected features, and computational efficiency. Furthermore, we investigate several major issues of memetic algorithm (MA) to identify a good balance between local search and genetic search so as to maximize search quality and efficiency in the hybrid filter and wrapper MA.**

***Index Terms*—Chi-square, feature selection, filter, gain ratio, genetic algorithm (GA), hybrid GA (HGA), memetic algorithm (MA), relief, wrapper.**

I. Introduction

Feature selection has become the focus of many research areas in recent years. With the rapid advance of computer and database technologies, data sets with hundreds and thousands of variables or features are now ubiquitous in pattern recognition, data mining, and machine learning [1]–[4]. To process such huge data sets is a challenging task because traditional machine learning techniques usually work well only on small data sets. Feature selection addresses this problem by removing the irrelevant, redundant, or noisy data. It improves the performance of the learning algorithm, reduces the computational cost, and provides better understandings of the data sets.

Feature selection algorithms may be widely categorized into two groups, i.e., 1) filter methods and 2) wrapper methods [2], [4]–[10]. Filter methods evaluate the goodness of the feature subset by using the intrinsic characteristic of the data. They are relatively computationally cheap since they do not involve the induction algorithm. However, they also take the risk of selecting subsets of features that may not match the chosen induction algorithm. Wrapper methods, on the contrary, directly use the induction algorithm to evaluate the feature subsets. They generally outperform filter methods in terms of prediction accuracy, but they are generally computationally more intensive. In summary, wrapper and filter methods can complement each other, in that filter methods can search through the feature space efficiently while the former provides good accuracy. In this correspondence, we propose a novel wrapper–filter feature selection algorithm (WFFSA) using a memetic framework [11]–[16], i.e., a combination of genetic algorithm (GA) [17]–[20] and local search (LS). Memetic algorithms (MAs) are population-based metaheuristic search methods inspired by Darwin's principles of natural evolution and Dawkins' notion of a meme defined as a unit of cultural evolution that is capable of local refinements. Recent studies on MAs have revealed their successes on a wide variety of real-world problems. Particularly, they not only converge to high-quality solutions but also search more efficiently than their conventional counterparts [11]–[16].

Manuscript received May 4, 2005; revised October 27, 2005 and January 9, 2006. This work was supported in part under the A*STAR SERC Grant 052 015 0024 administered through the National Grid Office. This paper was recommended by Associate Editor W.-Y. Wang.

The authors are with the Division of Information Systems, School of Computer Engineering, Nanyang Technological University, Singapore 639798 (e-mail: zhuzexuan@pmail.ntu.edu.sg; asysong@ntu.edu.sg; asmdash@ntu.edu.sg).

Digital Object Identifier 10.1109/TSMCB.2006.883267

```
Procedure of WFFSA
1  Begin
2     Initialize: Randomly generate an initial population of feature subsets;
3     While (Stopping conditions are not satisfied)
4        Evaluate all feature subsets encoded in the population;
5        For each subset chosen to undergo the local improvement process
6           Perform local search and replace it with locally improved
            solution in the spirit of Lamarckian learning;
7        End For
8        Perform evolutionary operators based on selection, crossover, and
         mutation;
9     End While
10 End
```

Fig. 1. Procedure of WFFSA.

The goal of WFFSA is to improve classification performance and accelerate the search to identify important feature subsets. In particular, the filter method fine-tunes the population of GA solutions by adding or deleting features based on univariate feature ranking information. Hence, our focus here is on filter methods that are able to assess the goodness or ranking of the individual features. We denote such filter methods as filter ranking methods in this correspondence and investigate the proposed WFFSA for several filter ranking methods. Empirical study of WFFSA on several commonly used data sets from the University of California, Irvine (UCI) repository [21] and several microarray data sets indicates that it outperforms recent existing methods in the literature in terms of classification accuracy, selected feature size, and efficiency. Furthermore, we also investigate the balance between LS and genetic search to maximize the search quality and efficiency of WFFSA.

The rest of this correspondence is organized as follows. Section II describes the WFFSA based on a memetic framework. The experimental results and discussions are presented in Section III. Finally, Section IV concludes this correspondence.

II. WFFSA

In this section, we introduce the proposed WFFSA for classification problems, which is depicted in Fig. 1. In the first step, the GA population is initialized randomly, with each chromosome encoding a candidate feature subset. Subsequently, on all or portions of the chromosomes, an LS or meme is applied in the spirit of Lamarckian learning [11]–[14]. The mechanism to do local improvement can be to reach a local optimum or to improve the solution. Genetic operators are then used to generate the next population. This process repeats until the stopping conditions are satisfied. We describe each component in detail as follows.

A. Encoding Representation and Initialization

In the feature selection problem, a representation for candidate feature subset must be chosen and encoded as a chromosome. In most

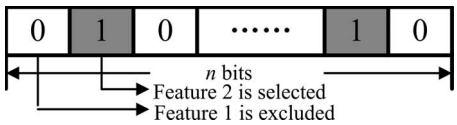

Fig. 2. Representation of chromosome as a binary bit string.

studies, a chromosome is a binary string of length equal to the total number of features, so that each bit encodes a single feature (as shown in Fig. 2). A bit of "1" ("0") implies the corresponding feature is selected (excluded). The length of the chromosome is denoted here as n. The maximum allowable number of bit 1's in each chromosome is denoted as m. When prior knowledge about the optimal number of features is available, we may limit m to no more than the predefined value; otherwise, m is equal to n. At the start of the search, a population size of p is randomly initialized.

B. Objective Function

The objective function is simply defined by the classification accuracy, i.e.,

$$\text{Fitness}\,(c) = J\,(S_c) \tag{1}$$

where S_c denotes the corresponding selected feature subset encoded in chromosome c, and the feature selection criterion function $J(S_c)$ evaluates the significance for the given feature subset S_c. In this correspondence, $J(S_c)$ is specified as the classification accuracy for S_c. Note that when two chromosomes are found having similar fitness, i.e., the difference between their fitness is less than a small value of ε, the one with the smaller number of selected features is given higher chances of surviving to the next generation.

C. LS Improvement Procedure

Much work on the use of domain knowledge and heuristics has resulted in highly effective search [11]–[13], [22]. Taking this cue, we consider here the use of filter ranking methods as memes or LS heuristics in our WFFSA. Later, we show in Section III that using filter ranking methods as memes, MA is capable of converging to improve classification accuracy and at lower number of selected features when compared to existing methods recently proposed in the literature.

Given a candidate chromosome c, we define X and Y as the sets of selected and excluded features encoded in c, respectively. Both X and Y are ranked using the univariate filter ranking method and with the most important feature ranked the highest. In this correspondence, we consider three different filter ranking methods, namely: 1) ReliefF [5]; 2) gain ratio [23]; and 3) chi-square [6]. These methods rank features based on different criteria that include Euclidean distance, information entropy, and chi-square statistics, respectively. We further define two basic LS operators of the WFFSA LS improvement procedure.

1) "Add": select a feature from Y using the linear ranking selection [24] and move it to X.
2) "Del": select a feature from X using the linear ranking selection [24] and move it to Y.

The Add and Del operations are illustrated in Fig. 3. Here, F5 and F4 are the highest and lowest ranked features in Y, whereas F3 and F6 are the highest and lowest ranked features in X. Using the Add and Del operations, F5 is the most likely feature to be moved to X, whereas F6 is the most likely feature to be moved to Y. The two most likely resultant chromosomes after the Add and Del operations are also depicted in Fig. 3.

The intensity of LS is quantified by the LS length l and interval w. LS length defines the maximum number of Del and Add operations

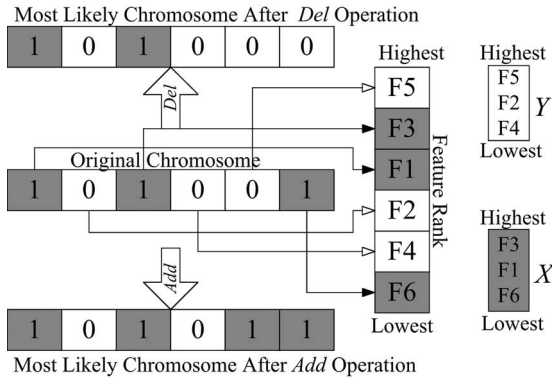

Fig. 3. Add and Del operations.

```
Procedure Improvement First Strategy
1  Begin
2     Initialize l and w;
3     For each chromosome c among the w elitists
4        For ( j = 1 to l^2 )
5           Generate a unique random pair (k,d);
6           Repeat k times of Add operation;
7           Repeat d times of Del operation;
8           Calculate fitness of improved chromosome c' using F(c')=J(c');
            //|•| denotes the cardinality of a vector
9           If ((F(c') > F(c)) or (| F(c') - F(c)| < ε and | c'| < |c|))
10             Replace the genotype c with the improved c';
11             Break and consider the next elite chromosome;
12          End If
13       End For
14    End For
15 End
```

Fig. 4. Procedure of improvement first strategy.

in each LS. Therefore, there are a total of l^2 possible combinations of Add and Del operations applied on a chromosome. LS interval specifies the w elite chromosomes in the population that undergo LS improvement procedure in each generation. The LS improvement procedure may be applied on a chromosome until a local optimum or an improvement is reached. The locally improved chromosome is evaluated and replaces the original chromosome if it is of higher quality. Here, we further investigate three different LS strategies that are characterized by different LS intensity.

1) Improvement First Strategy: In this strategy, a random choice from the l^2 combinations of Del and Add operations is used to search on the candidate chromosome. The LS stops once an improvement is obtained either in terms of classification accuracy or a reduction in the number of selected features without deterioration in accuracy greater than ε. This procedure is outlined in Fig. 4.

2) Greedy Strategy: In contrast to the improvement first strategy, the greedy strategy carries out all possible l^2 combinations of Del and Add operations, and the best improved solution is used to replace the original chromosome in the population. The greedy strategy is outlined in Fig. 5.

3) Sequential Strategy: We also consider the sequential strategy described in [16]. The hybrid GA (HGA) was reported in [16] to generate better search performances than GA, sequential forward search (SFS), sequential forward floating search (SFFS), PTA(l, r) (plus—l and take away—r) and multistart algorithms. There, instead of using a filter ranking method, the Add operation searches for the most significant feature y in Y in a sequential manner, i.e., $y = \arg\max_{a \in Y} J\,(X \cup \{a\})$, and moves it to X. In the same way, the Del operation searches for the least significant feature x from X in

```
Procedure Greedy Strategy
Begin
    Initialize l and w;
    For each chromosome c among the w elitists
        c_best = c;
        For each of the l^2 combinations (k,d)
            Repeat k times of Add operation;
            Repeat d times of Del operation;
            Calculate fitness of improved chromosome c' using F(c')=J(c');
            If ((F(c') > F(c_best)) or (| F(c') - F(c_best)| < ε and | c'| < |c_best|))
                c_best = c'; //update the best improved chromosome
            End If
        End For
        Replace the genotype c with the best improved c_best;
    End For
End
```

Fig. 5. Procedure of greedy strategy.

```
Procedure Restrictive Crossover
Begin
    Randomly select two parents p1 and p2;
    Randomly generate a number r within [0,1];
    If (r < pc)      //pc denotes the crossover probability
        //ensure the number of bit '1' in the offspring dose not exceed m
        k = Min( |p1| , |p2| );      // |p1|, |p2| < m
        For ( i = 1 to k )
            Locate the allele L1 of the i^th bit '1' in p1;
            Locate the allele L2 of the i^th bit '1' in p2;
            Crossover p1 and p2 in positions L1 and L2 with probability 0.5;
        End For
    End If
End
```

Fig. 6. Procedure of restrictive crossover.

```
Procedure Restrictive Mutation
Begin
    // mutating chromosome c; pm denotes the mutation probability
    For ( i = 1 to |c| )
        Locate the position L1 of the i^th bit '1' in c;
        Randomly select a bit '0' with position L0;
        Swap positions L1 and L0 with probability pm;
    End For
    For ( i = 1 to m-|c|)
        Randomly flip a bit '0' with probability pm;
    End For
End
```

Fig. 7. Procedure of restrictive mutation.

a sequential manner, i.e., $x = \arg\max_{a \in X} J(X - a)$, and moves it to Y.

4) Evolutionary Operators: In the evolution process, standard GA operators such as linear ranking selection, uniform crossover, and mutation operators based on elitist strategy may be applied. However, if prior knowledge on the optimum number of features is available, the number of bit 1 in each chromosome may be constrained to a maximum of m in the evolution process. Since the standard uniform crossover and mutation operators may violate this constraint, restrictive crossover and mutation are proposed here. In restrictive crossover, the crossover operations are applied only on alleles with bit 1 and in either of the two parent chromosomes. We outline the restrictive crossover in Fig. 6. Based on the same principles, the restrictive mutation operator is outlined in Fig. 7.

5) Computational Complexity: In this section, we analyze the computational complexity of the proposed WFFSA. The ranking of features based on the filter methods have linear time complexity in terms of feature dimensionality: they are conducted offline, and the obtained rank list may be reused for each LS in WFFSA. Consequently, the computational for feature ranking is a one-time offline cost and is considered to be negligible compared to that of fitness evaluation in (1). Hence, we define the computational cost of a single fitness evaluation as the basic unit of computational cost in our analysis.

The computational complexity for GA can be derived as $O(pg)$, where p is the size of population and g is the number of search generations. The expected computational complexity of WFFSA with improvement first strategy is $O(pg + l^2 wg/2)$. Here, we assume that each l^2 combination of Add and Del operation has equivalent likelihood to obtain the first improvement; therefore, the average trails to obtain the first improvement is $l^2/2$. In a single search generation, $l^2 w/2$ fitness function calls are incurred in the LS. The ratio of LS to genetic search is thus $l^2 w/2p$.

In the greedy strategy, LS evaluates all possible l^2 combinations of Add and Del operations to attain the best possible locally improved solution. The computational complexity is thus $O(pg + l^2 wg)$. In a single search generation, $l^2 w$ fitness function calls are incurred in the LS, and the ratio of LS to genetic search is $l^2 w/p$.

For the sequential strategy proposed in HGA [16], a total of $(2|Y| - l)l/2$ and $(2|X| + l)l/2$ calls to the fitness function are incurred on the Add and Del operations, respectively. Since $|X| + |Y| = n$, the total classification calls in the LS is nlw. The computational complexity of sequential strategy is thus $O(pg + nlwg)$. The ratio of LS to genetic search is then nlw/p.

Since we are working with classification problems where $n \gg l$, it is easy to determine that $nlw \gg l^2 w > l^2 w/2$. Consequently, the sequential LS strategy in HGA [16] would require significantly more computations, hence incurring more time than both the improvement first strategy and greedy strategy of the WFFSA.

III. Experimental Results and Discussions

In this section, we present an experimental study of WFFSA on commonly used benchmark and biological data sets. In particular, we focus on four UCI data sets having more than 15 features and four microarray data sets (i.e., ALL/AML [25], Colon [26], NCI60 [27], and SRBCT [28]) having thousands of features in our study.

In the WFFSA, we employed a population size of 30 and a stopping criterion of 6000 fitness function calls for UCI data sets. On microarray data sets, as the feature size is significantly larger, the population size is increased to 50 or 100, and the stopping criterion is increased to 10 000 or 20 000. Furthermore, the maximum number of selected features is unconstrained for the UCI data sets, i.e., $m = n$. On the other hand, it has been shown in the literature [8], [25]–[29] that microarray data sets could be learned with high accuracies with only hundreds or tens of features. In effect, we make use of this prior knowledge to constrain the maximum number of selected features m as depicted in Table I. In effect, the restrictive crossover and mutation operations are applied on the microarray data sets. In our experimental setup, we employ crossover and mutation probabilities of $p_c = 0.6$ and $p_m = 0.1$, respectively. Linear ranking selection [24] with selection pressure of 1.5 is used for selection. The threshold ε to determine fitness similarity between two chromosomes is configured as 0.001 for UCI data sets and 0.02 for microarray data sets. The fitness of a chromosome or selected feature subset is evaluated using the one nearest neighbor (1NN) classifier and the leave-one-out cross validation (LOOCV). Here, we use the classification accuracy estimated from LOOCV and the number of selected features as performance measures. It is worth noting that the configurations of the parameter used here have been

TABLE I
DATA SETS AND PARAMETERS USED FOR EXPERIMENTS

Dataset	Num of Features, n	Num of Instances	Num of Classes	Population Size	p_c , p_m	Maximum Number of Selected Features, m	Feature Subset Evaluation	Stopping Criterion *
Vehicle	18	846	4	30	0.6, 0.1	18	1NN, LOOCV	6000
WDBC	30	569	2	30	0.6, 0.1	30	1NN, LOOCV	6000
Ionosphere	34	351	2	30	0.6, 0.1	34	1NN, LOOCV	6000
Sonar	60	208	2	30	0.6, 0.1	60	1NN, LOOCV	6000
ALL/AML	1000	72	2	50	0.6, 0.1	Constrained to 50	1NN, LOOCV	10000
Colon	1000	62	2	50	0.6, 0.1	Constrained to 50	1NN, LOOCV	10000
NCI60	1000	60	9	50	0.6, 0.1	Constrained to 50	1NN, LOOCV	10000
SRBCT	2308	83	4	100	0.6, 0.1	Constrained to 50	1NN, LOOCV	20000

* Here the stopping criterion is the maximum number of fitness function calls

TABLE II
COMPARISON RESULTS OF FILTER, GA, AND WFFSA METHODS

Dataset	Best Found	ReliefF	Gain Ratio	Chi Square	GA	WFFSA-R*		WFFSA-G*		WFFSA-C*	
						First**	Greedy**	First	Greedy	First	Greedy
Vehicle (846×18)	***75.06, 12***	72.81, 12	70.33, 16	72.22, 15	74.88, 10.2 ***(75.06, 12)***	75.00, 11.3 ***(75.06, 12)***	**75.06, 12** ***(75.06, 12)***	74.98, 11.5 ***(75.06, 12)***	75.04, 11.7 ***(75.06, 12)***	**75.06, 12** ***(75.06, 12)***	75.04, 11.7 ***(75.06, 12)***
WDBC (569×30)	***98.24, 12***	96.49, 4	96.31, 29	96.31, 29	97.82, 14.7 (98.23, 19)	97.96, 13 (98.24, 14)	**98.14, 13.9** (98.24, 14)	97.98, 11.8 (98.24, 14)	97.93, 13.2 (98.07, 11)	97.88, 12 (98.24, 15)	98.00, 12.7 ***(98.24, 12)***
Ionosphere (351×34)	***96.01, 8***	91.74, 7	90.88, 13	92.31, 7	94.13, 12.5 (94.87, 13)	95.00, 7.5 (95.73, 8)	94.96, 9.7 (95.44, 8)	95.13, 8.9 ***(96.01, 8)***	**95.19, 8.9** (95.44, 7)	95.01, 8.9 (95.44, 7)	94.73, 10.4 (95.16, 9)
Sonar (208×60)	***97.6, 22***	88.46, 15	88.94, 31	89.90, 30	93.89, 27.7 (95.19, 25)	**96.30, 24** (97.12, 19)	95.63, 24.6 (96.63, 24)	94.95, 25.2 ***(97.6, 22)***	95.22, 24 (96.63, 24)	95.82, 24.6 (96.63, 24)	95.24, 26.4 (96.15, 26)
ALL/AML (72×1000)	***100, 2***	98.61, 16	98.61, 17	98.61, 18	100, 26.5 (100, 20)	**100, 5.1** ***(100, 2)***	100, 8.3 (100, 4)	100, 5.5 (100, 3)	100, 6.3 (100, 4)	100, 6.1 ***(100, 2)***	100, 7.9 (100, 3)
Colon (62×1000)	***100, 14***	83.87, 17	90.32, 24	87.10, 17	94.52, 37 (95.16, 31)	**97.9, 10.9** ***(100, 14)***	97.26, 16 (100, 20)	96.29, 11.9 (98.39, 11)	97.9, 14.9 (100, 17)	95.97, 10 (98.39, 15)	96.94, 17.8 (100, 16)
NCI60 (60×1000)	***85.25, 14***	65.57, 16	59.02, 31	65.57, 28	77.54, 38.6 (80.33, 40)	**82.30, 22.7** (85.25, 19)	79.67, 23.8 (81.97, 24)	82.13, 21.5 (85.25, 18)	81.31, 27.5 (85.25, 33)	81.15, 21.8 ***(85.25, 14)***	81.48, 26.5 (85.25, 22)
SRBCT (83×2308)	***100, 7***	100, 38	100, 20	100, 43	99.64, 43.3 (100, 39)	**100, 15.1** ***(100, 7)***	100, 18.7 (100, 11)	99.76, 17.1 (100, 8)	100, 21.8 (100, 16)	99.76, 14.9 (100, 9)	99.88, 17.9 (100, 8)

The value delimited by comma in each grid shows the classification accuracy and the corresponding number of selected features respectively. The values in the parentheses are the best results obtained. Bold typefaces emphasize the best average performance in each row. Bold italic typefaces emphasize the best solution found among all the methods. * Here WFFSA-R, WFFSA-G, and WFFSA-C denote WFFSA with local search heuristic of ReliefF, Gain Ratio, and Chi-Square, respectively. ** First represents the improvement first strategy with length $l = 4$ and interval $w = 1$.Greedy represents greedy strategy with $l = 4$ and $w = 5$.

investigated empirically for the data sets considered and are summarized in Table I.

A. Comparison of Filter, GA, and WFFSA

The search performances of the filter ranking methods GA and WFFSA using several filter ranking methods and LS strategies on the eight data sets considered are summarized in Table II. The comma-delimited pairwise numeric values in the table represent the classification accuracy and the corresponding number of selected features. Due to the stochastic nature of GA and WFFSA, the average results for ten independent runs are reported. The parentheses highlight the best result found across the ten runs. For each data set, the bold-faced and bold-italic-faced representations in Table II emphasize the best average performance and the best solution found among all methods, respectively. Six WFFSAs obtained based on the three filter ranking methods (i.e., ReliefF—WFFSA-R, gain ratio—WFFSA-G, or chi-square—WFFSA-C) and two LS strategies (i.e., improvement first or greedy) have been investigated. In the WFFSAs, the LS length l and interval w in improvement first strategy and greedy strategy are configured as $l = 4$, $w = 1$ and $l = 4$, $w = 5$, respectively.

We discuss the performances for the various feature selection algorithms considered. On filter ranking methods, the t features with highest rank are chosen to induce the 1NN LOOCV classification. We increase t from 1 to m, and the optimal value is determined when the best classification accuracy is obtained. The best classification accuracy and the corresponding t value for each filter ranking methods are reported in Table II (i.e., columns 3–5). It can be observed that WFFSAs outperform all three filter ranking methods and GA in terms of classification accuracy. The improvement in performance is more significant for the Sonar, Colon, and NCI60 data sets. Moreover, WFFSAs reduce the number of selected features significantly. On the ALL/AML and SRBCT data sets, WFFSAs use less than one-third of the features required by GA and filter ranking methods to arrive at the improved classification accuracy. The best solutions found on all eight data sets shown in column 2 were attained by WFFSAs. We also study the performance of WFFSA for three different filter ranking methods since it was shown in [11] that inappropriate use of meme may result in MA performing poorer than standard GA. The results in Table II indicate that WFFSA-R performs better than the other counterparts on seven out of the eight data sets in terms of average performance. For ReliefF filter ranking method, the feature goodness is evaluated by its ability to distinguish the near hit (nearest neighbors from the same class) and near miss (nearest neighbors from different classes). Therefore, it makes good sense that features with high score in ReliefF are more likely to help 1NN identify the correct nearest neighbor, hence generating good classification accuracy. On the other hand, gain ratio and chi-square do not appear to possess such mechanisms to compliment 1NN, hence being less effective here.

Furthermore, we compare the results obtained by WFFSAs to the recently proposed HGA(3) [16] for the UCI data sets in Table III. The best average performance and the best solution for each data set are highlighted using bold typeface representation. The results indicate that WFFSAs generate the best solutions on all four data sets

TABLE III
RESULTS BY WFFSAs AND OTHER METHODS ON UCI DATA SETS

Dataset	WFFSA		Results obtained from the literature	
Vehicle	**75.06, 12** ***(75.06, 12)***	WFFSA+1NN LOOCV	73.52,7 *(73.52, 7)*	HGA(3)+1NN LOOCV [16]
WDBC	**98.14, 13.9** ***(98.24, 12)***	WFFSA+1NN LOOCV	93.85, 24 *(93.85, 24)*	HGA(3)+1NN LOOCV [16]
Ionosphere	95.19, 8.9 ***(96.01, 8)***	WFFSA+1NN LOOCV	**95.56, 7** *(95.73, 7)*	HGA(3)+1NN LOOCV [16]
Sonar	96.30, 24 ***(97.6, 22)***	WFFSA+1NN LOOCV	**96.34, 24** *(97.12, 24)*	HGA(3)+1NN LOOCV [16]

Bold and bold italic typefaces represent best average performance and best solution found among the methods, respectively.

TABLE IV
RESULTS BY WFFSAs AND OTHER METHODS ON MICROARRAY DATA SETS

Dataset	WFFSA		Results obtained from the literature	
ALL/AML	**97.34, 30.5** ***(98.57, 30.8)***	WFFSA+SVM 10-fold CV	97.34, 31 *(97.68, 50)*	SVM+LS bound .632+ bootstrap [25]
	100, 5.1 ***(100, 2)***	WFFSA+1NN LOOCV	*(100, 8)*	SVM-RFE Holdout [8]
Colon	**86.01, 31.0** ***(87.38, 31.2)***	WFFSA+SVM 10-fold CV	84.77, 31 *(84.95, 46)*	SVM+LS bound .632+ bootstrap [25]
	97.9, 10.9 ***(100, 14)***	WFFSA+1NN LOOCV	*(100,16)*	SVM-RFE Holdout [8]
SRBCT	98.53, 90.5 ***(100, 89.6)***	WFFSA+SVM 10-fold CV	*(~95.00, 150)*	SVM+Max Minority 4-fold CV [29]
	100, 15.1 ***(100, 7)***	WFFSA+1NN LOOCV	*(100, 96)*	ANN+PCA 3-fold CV [28]
NCI60	66.43, 485.3 ***(72.19, 485)***	WFFSA+SVM 4-fold CV	*(66.66, 150)*	SVM+Sum Minority 4-fold CV [29]
	82.30, 22.7 *(85.25, 14)*	WFFSA+1NN LOOCV	***(85.25,13)***	GA+MLHD LOOCV [27]

Bold and bold italic typefaces represent best average performance and best solution found among the methods, respectively. Non-shaded and shaded tables represent using Internal or External cross validation, respectively. External cross-validation is repeated 10 times for each dataset.

and give competitive results to existing methods in terms of average performance. Even so, it is worth noting that HGA(3) consumed more than 200 000 fitness function calls to arrive at the competitive performance on these UCI data sets. In contrast, WFFSA incurs less than 6000 fitness function calls to arrive at superior or competitive performances.

To illustrate the generality and efficacy of the WFFSA framework, we consider also the use of different induction algorithms, particularly standard 1NN and support vector machine (SVM) with radius margin bound [30], and compare their performances with other recent studies in the literature using internal and external cross-validation schemes on the real-world microarray data sets. In external cross validation, each data set is randomly split into k stratified folds with $k-1$ folds for training and one fold for testing. The performance of the final selected features obtained is measured on the unseen testing data. The procedure is repeated k times. The best average performance and/or the best solution are reported in Table IV. Table IV indicates that WFFSAs display superior performances on most of the microarray data sets in comparison to the existing counterparts.

B. Study on LS Strategies

Over the recent years, much work has shown that an appropriate balance between LS and genetic search is necessary for efficient memetic search [11]–[15]. In this section, we examine the balance between LS and genetic search using different LS strategies, which are characterized by different intensities of genetic search and LS. Ten independent runs of WFFSA with different combinations of strategy, LS length l, and interval w are conducted on two representative UCI and microarray data sets, i.e., the Sonar and Colon data sets. The results obtained are summarized in Tables V and VI. The LS heuristic or filter ranking method used is ReliefF since it is shown to give better search performance than the other counterparts.

TABLE V
RESULTS OF DIFFERENT LS STRATEGIES, LENGTH, AND INTERVAL ON SONAR DATA SET

	Improvement First Strategy			Greedy Strategy		
	$w=1$	$w=5$	$w=P$	$w=1$	$w=5$	$w=P$
$l=2$	95.77, 26.3	94.95, 26.7	95.10, 26.1	94.86, 28.3	95.14, 25.3	94.52, 28.3
$l=4$	**96.30, 24**	95.67, 24.7	95.38, 22.9	94.8, 23.2	95.63, 24.6	94.13, 26
$l=8$	95.53, 21.4	95.24, 24.4	95.43, 25.5	94.91, 20.5	95.10, 25	94.04, 28.1

TABLE VI
RESULTS OF DIFFERENT LS STRATEGIES, LENGTH, AND INTERVAL ON COLON DATA SET

	Improvement First Strategy			Greedy Strategy		
	$w=1$	$w=5$	$w=P$	$w=1$	$w=5$	$w=P$
$l=2$	97.26, 17.9	97.10, 16.9	95.97, 22.2	96.13, 15	96.29, 14.7	95.65, 20.8
$l=4$	**97.9, 11.5**	97.42, 12.9	97.10, 14.6	96.45, 12.6	97.26, 16	95.65, 19.9
$l=8$	96.77, 10.5	97.26, 10.4	97.10, 13.1	96.45, 10.9	96.94, 14.1	95,32, 15.8

TABLE VII
RESULTS OF DIFFERENT LOCAL STRATEGIES ON ALL EIGHT DATA SETS

Dataset	WFFSA-R First $w=1$, $l=4$	WFFSA-R Greedy $w=5$, $l=4$	MA-S* $w=1$, $l=4$	MA-S* $w=5$, $l=4$
Vehicle (846×18)	75.00, 11.25 ***(75.06, 12)***	**75.06, 12** ***(75.06, 12)***	74.84, 10.5 ***(75.06, 12)***	74.60, 10.6 ***(75.06, 12)***
WDBC (569×30)	97.96, 13 ***(98.24, 14)***	**98.14, 13.9** ***(98.24, 14)***	97.80, 14.7 (98.07, 14)	97.47, 13.7 (97.72, 12)
Ionosphere (351×34)	**95.00, 7.5** ***(95.73, 8)***	94.96, 9.7 (95.44, 8)	94.27, 11 (95.44, 10)	93.48, 12.8 (94.59, 12)
Sonar (208×60)	**96.30, 24** ***(97.12, 19)***	95.63, 24.6 (96.63, 24)	93.65, 26.5 (95.67, 25)	91.73, 28.4 (93.75, 32)
ALL/AML (72×1000)	**100, 5.1** ***(100, 2)***	100, 8.3 (100, 4)	100, 27.2 (100, 16)	100, 44 (100, 44)
Colon (62×1000)	**97.9, 11.5** ***(100, 14)***	97.26, 16 (100, 20)	91.94, 31.4 (95.16, 19)	82.26, 18 (82.26, 18)
NCI60 (60×1000)	**82.30, 22.7** ***(85.25, 19)***	79.67, 23.8 (81.97, 24)	74.36, 36 (77.05, 36)	68.85, 34 (68.85, 34)
SRBCT (83×2308)	**100, 15.1** ***(100, 7)***	100, 18.7 (100, 11)	93.13, 39 (96.39, 33)	87.83, 35.5 (91.57, 23)

* MA-S denotes memetic algorithm based on sequential local search strategy.

The results in Tables V and VI indicate that WFFSA for $l=4$ and $w=1$ obtains the best average accuracy on the improvement first strategy. On the Sonar data set, WFFSA for $l=4$ and $w=1$ and using the improvement first strategy is found to perform significantly better than the other configurations of WFFSAs statistically, using the two-tailed paired t-test at significance level of 0.05. On the Colon data set, the superiority of WFFSA for $l=4$, $w=1$ and an improvement first strategy is, however, not statistically conclusive using the same t-test. Overall, we note that WFFSA with improvement first strategy generally performs better than using the greedy strategy. WFFSA obtains better results when LS is applied only to the single or five elite chromosomes instead of the entire population. In line with the observations in [12] and [14], conducting LS on all chromosomes can result in unnecessary computation. Since the computational budget is often limited (i.e., based on the maximum number of fitness function calls allowed), the genetic search will be reduced proportionally due to possible redundant computation spent on LS. Without sufficient genetic search, the MA is more likely to be trapped in the local optimum and hence may lead to poor search quality under limited computational budget.

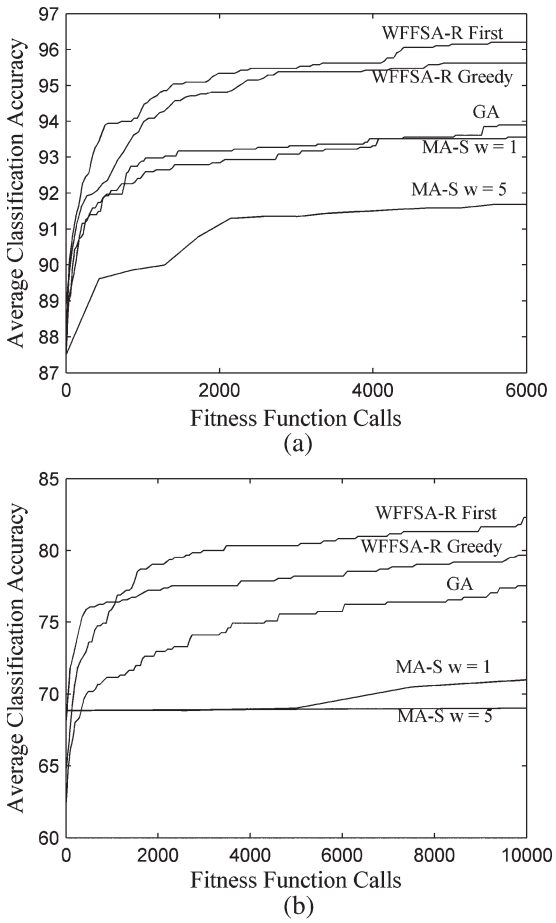

Fig. 8. Search trace (average of ten independent runs) of GA, WFFSA-R First ($l = 4$, $w = 1$), WFFSA-R Greedy ($l = 4$, $w = 5$), MA-S ($l = 4$, $w = 1$), and MA-S ($l = 4$, $w = 5$) on (a) Sonar data set and (b) NCI data set.

We also compare WFFSA-R with an MA based on sequential strategy (MA-S). In MA-S, we configure $l = 4$ and $w = 5$. The comparison results are reported in Table VII. On all the data sets, both WFFSA-R with improvement first strategy or greedy strategy (they are labeled as WFFSA-R First and WFFSA-R Greedy, respectively, in Table VII) displays significantly better performance than MA-S statistically using two-tailed paired t-test at significance level of 0.05. The superior performance of WFFSA over MA-S is more obvious on the microarray data sets where $n \gg l$. So far, the results obtained further strengthen the importance of balance tradeoff between LS and genetic search for efficient MA search. A search dominated by either genetic search (e.g., GA) or LS (MA-S) would generally not perform effectively. The WFFSA for $l = 4$ and $w = 1$ based on the improvement first strategy thus gives the most appropriate tradeoff of LS and genetic search than GA, MA-S, and all other configurations considered in this correspondence. As shown in Fig. 8, the average search trends of WFFSA-R First, WFFSA-R Greedy, MA-S ($l = 4$, $w = 1$), and MA-S ($l = 4$, $w = 5$) on the Sonar and NCI data sets suggest that WFFSA-Rs search more efficiently than GA and MA-S under limited computational budget.

IV. Conclusion

In this correspondence, we have proposed a novel hybrid filter and wrapper feature selection algorithm based on a memetic framework. We use filter ranking method as LS heuristic in the MA. The experimental results presented show that the proposed method searches more efficiently and is capable of producing good classification accuracy with a small number of features simultaneously. Most importantly, it outperforms GA, MA with sequential LS, as well as many existing algorithms in the literature. Furthermore, our study on various LS strategies, LS length, and interval allows us to identify a suitable balance tradeoff of genetic search and LS in the memetic search. This allows us to maximize the effectiveness and efficiency of the proposed hybrid filter and wrapper feature selection algorithm for classification problem using a memetic framework.

References

[1] A. Jain and D. Zongker, "Feature selection: Evaluation, application, and small sample performance," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 2, pp. 153–158, Feb. 1997.

[2] M. Dash and H. Liu, "Feature selection for classification," *Int. J. Intell. Data Anal.*, vol. 1, no. 3, pp. 131–156, 1997.

[3] ——, "Consistency-based search in feature selection," *Artif. Intell.*, vol. 151, no. 1/2, pp. 155–176, 2003.

[4] R. Kohavi and G. H. John, "Wrapper for feature subset selection," *Artif. Intell.*, vol. 97, no. 1/2, pp. 273–324, 1997.

[5] M. Robnic-Šikonja and I. Kononenko, "Theoretical and empirical analysis of ReliefF and RReliefF," *Mach. Learn.*, vol. 53, no. 1/2, pp. 23–69, 2003.

[6] H. Liu and R. Setiono, "Chi2: Feature selection and discretization of numeric attributes," in *Proc. 7th IEEE Int. Conf. Tools With Artif. Intell.*, 1995, pp. 388–391.

[7] E. Xing, M. Jordan, and R. Karp, "Feature selection for high-dimensional genomic microarray data," in *Proc. 15th Int. Conf. Mach. Learn.*, 2001, pp. 601–608.

[8] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Mach. Learn.*, vol. 46, no. 1–3, pp. 389–422, 2002.

[9] K. Z. Mao, "Feature subset selection for support vector machines through discriminative function pruning analysis," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 1, pp. 60–67, Feb. 2004.

[10] C. N. Hsu, H. J. Huang, and S. Dietrich, "The ANNIGMA–wrapper approach to fast feature selection for neural nets," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 32, no. 2, pp. 207–212, Apr. 2004.

[11] Y. S. Ong and A. J. Keane, "Meta-Lamarckian in memetic algorithm," *IEEE Trans. Evol. Comput.*, vol. 8, no. 2, pp. 99–110, Apr. 2004.

[12] Y. S. Ong, M. H. Lim, N. Zhu, and K. W. Wong, "Classification of adaptive memetic algorithms: A comparative study," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 36, no. 1, pp. 141–152, Feb. 2006.

[13] H. Ishibuchi, T. Yoshida, and T. Murata, "Balance between genetic search and local search in memetic algorithm for multiobjective permutation flowshop scheduling," *IEEE Trans. Evol. Comput.*, vol. 7, no. 2, pp. 204–223, Apr. 2003.

[14] N. Krasnogor, "Studies of the theory and design space of memetic algorithms," Ph.D. dissertation, Univ. West England, Bristol, U.K., 2002.

[15] Q. Zhang, J. Sun, E. Tsang, and J. Ford, "Hybrid estimation of distribution algorithm for global optimization," *Eng. Comput.*, vol. 21, no. 1, pp. 91–107, 2004.

[16] I. S. Oh, J. S. Lee, and B. R. Moon, "Hybrid genetic algorithm for feature selection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 11, pp. 1424–1437, Nov. 2004.

[17] W. Siedelecky and J. Sklansky, "On automatic feature selection," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 2, no. 2, pp. 197–220, 1988.

[18] M. L. Raymer, W. F. Punch, E. D. Goodman, L. A. Kuhn, and A. K. Jain, "Dimensionality reduction using genetic algorithms," *IEEE Trans. Evol. Comput.*, vol. 4, no. 2, pp. 164–171, Jul. 2000.

[19] L. Li, C. R. Weinberg, T. A. Darden, and L. G. Pedersen, "Gene selection for sample classification based on gene expression data: Study of sensitivity to choice of parameters of the GA/KNN method," *Bioinformatics*, vol. 17, no. 12, pp. 1131–1142, 2001.

[20] J. H. Yang and V. Honavar, "Feature selection using a genetic algorithm," *IEEE Intell. Syst.*, vol. 13, no. 2, pp. 44–49, Mar./Apr. 1998.

[21] D. J. Newman, S. Hettich, C. L. Blake, and C. J. Merz, (1998). *UCI repository of machine learning databases*. Irvine, CA: Dept. Inf. Comput. Sci., Univ. California. [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html

[22] Y. S. Ong and A. J. Keane, "A domain knowledge based search advisor for design problem solving environments," *Eng. Appl. Artif. Intell.*, vol. 15, no. 1, pp. 105–116, 2002.

[23] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann, 1993.

[24] J. E. Baker, "Adaptive selection methods for genetic algorithms," in *Proc. Int. Conf. Genetic Algorithm and Their Appl.*, 1985, pp. 101–111.
[25] X. Zhou and K. Z. Mao, "LS bound based gene selection for DNA microarray data," *Bioinformatics*, vol. 21, no. 8, pp. 1559–1564, 2005.
[26] U. Alon, N. Barkai, D. Notterman, K. Gish, S. Ybarra, D. Mack, and A. Levine, "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays," *Proc. Nat. Acad. Sci. USA*, vol. 96, no. 12, pp. 6745–6750, 1999.
[27] C. H. Ooi and P. Tan, "Genetic algorithms applied to multi-class prediction for the analysis of gene expression data," *Bioinformatics*, vol. 19, no. 1, pp. 37–44, 2003.
[28] J. Khan, J. S. Wei, M. Ringer, L. H. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C. R. Antonescu, C. Peterson, and P. S. Meltzer, "Classification and diagnostic prediction of cancers using expression profiling and artificial neural networks," *Nat. Med.*, vol. 7, no. 6, pp. 673–679, 2001.
[29] T. Li, C. Zhang, and M. Ogihara, "A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression," *Bioinformatics*, vol. 20, no. 15, pp. 2429–2437, 2004.
[30] O. Chapelle, V. Vapnik, O. Bousqet, and S. Mukherjee, "Choosing multiple parameters for support vector machines," *Mach. Learn.*, vol. 46, no. 1, pp. 131–159, 2002.