# Postprint of Gene Expression Data Classification Algorithm Based on FCBF Feature Selection and Ensemble Optimization Learning

**Authors:** Ma Chao

**Date:** 2018-07-09T00:00:00+00:00

## Abstract

To address the problems of high dimensionality, small sample size, high redundancy, and high noise in microarray gene expression data, we propose a classification algorithm based on FCBF feature selection and ensemble optimized learning, termed FICS-EKELM. First, the fast correlation-based filter (FCBF) method is utilized to filter out partially irrelevant features and noise, identifying a feature set with high class correlation. Second, sampling techniques are employed to generate multiple sample subsets. On each training subset, an improved crow search algorithm is leveraged to simultaneously achieve optimal feature subset selection and parameter optimization for the kernel extreme learning machine (KELM) classifier. Then, an ensemble classification model is constructed based on base classifiers to perform classification and identification on target data. Furthermore, a multi-threaded parallel approach on a multicore platform is adopted to further improve the computational efficiency of the algorithm. Experimental results on six gene datasets show that the proposed method not only achieves superior classification performance with fewer feature genes, but also produces classification results significantly better than existing and similar methods, making it an effective approach for high-dimensional data classification.

## Full Text

### Gene Expression Data Classification Based on FCBF Feature Selection and Ensemble Optimized Learning

**College of Digital Media, Shenzhen Institute of Information Technology, Shenzhen, Guangdong 518172, China**

**Abstract**

To address the challenges of high dimensionality, small sample size, high redundancy, and significant noise in microarray gene expression data, this paper proposes a novel classification algorithm called FICS-EKELM that combines FCBF feature selection with ensemble optimized learning. The method first employs the Fast Correlation-Based Filter (FCBF) method to eliminate irrelevant features and noise, identifying feature subsets with high correlation to class labels. Second, sampling techniques generate multiple training subsets, upon each of which an improved crow search algorithm is used to simultaneously select optimal feature subsets and optimize parameters for the Kernel Extreme Learning Machine (KELM) classifier. An ensemble classification model is then constructed based on these base classifiers for target data classification, with computational efficiency further enhanced through multi-threaded parallel processing on multi-core platforms. Experimental results on six gene datasets demonstrate that the proposed method not only achieves superior classification performance with fewer feature genes but also significantly outperforms existing and comparable methods, proving its effectiveness for high-dimensional data classification.

**Keywords:** feature selection; ensemble learning; microarray gene expression data; crow search algorithm; kernel extreme learning machine

## 0 Introduction

DNA microarray technology represents a significant technical breakthrough in bioinformatics that has been widely applied in drug research, gene sequencing, and other fields. The gene expression data obtained through microarray technology, typically represented in matrix form, analyzes changes in genes, their interrelationships, and resulting impacts. Classification constitutes a crucial task in microarray gene expression data mining, as analyzing such data can provide reliable classification results for disease diagnosis and treatment. However, microarray gene expression data suffers from the "curse of dimensionality" — high dimensionality with small sample sizes—which renders traditional pattern classification research difficult to resolve. Consequently, effective feature selection and classification to identify the small subset of genes most contributory to classification and improve classification performance has become a key challenge in gene expression data classification research.

In recent years, systematic analysis of gene expression data has emerged as a 热门研究课题 in artificial intelligence. Numerous dimensionality reduction methods have been applied to feature selection and recognition in gene data. Among these, gene feature selection methods represent the primary approach for dimensionality reduction and can be categorized based on their independence from subsequent learning algorithms into Filter and Wrapper methods:

a) **Filter Methods**: These approaches generally employ evaluation criteria to enhance feature-class correlation while reducing inter-feature correla-

tion, utilizing metrics such as Information Gain (IG), minimum Redundancy Maximum Relevance (mRMR), ReliefF, FCBF, Fisher Score, and minimum squared regression error. Filter methods operate independently of subsequent learning algorithms, typically leveraging statistical properties of all training data to evaluate features. However, they fail to consider correlations between features and classifiers, cannot guarantee selection of an optimal feature subset, and even when finding a satisfactory subset, tend to produce relatively large feature sets containing obvious noise features, resulting in significant performance deviation from subsequent learning algorithms.

b) **Wrapper Methods**: These approaches function as integral components of learning algorithms, directly using classifier performance as the evaluation criterion for feature importance and constructing final classification models based on selected feature subsets. Though slower than Filter methods, Wrapper methods produce considerably smaller feature subsets that facilitate feature identification and achieve higher classification accuracy. Recognizing these complementary characteristics, many current studies employ hybrid Filter-Wrapper approaches.

For instance, Jerzy et al. [7] in 2016 utilized ReliefF and mRMR combined with SVM for high-dimensional cancer data classification, achieving favorable results. In the same year, Xie Juanying et al. [8] proposed a hybrid method based on K-S test and mRMR principles, using SVM as the classifier and evaluating gene selection through F1_measure, classification accuracy, and AUC, with results demonstrating method effectiveness. Lai et al. [9] combined Filter and Wrapper methods, proposing Information Gain (IG) and Improved Simplified Swarm Optimization (ISSO) with a linear kernel SVM classifier for gene feature selection. Lu et al. [10] in 2017 proposed a hybrid feature selection algorithm combining Maximal Information Maximization (MIM) and adaptive genetic algorithms to reduce gene expression data dimensionality. Wang et al. [11] addressed the dimensionality disaster in gene expression data by proposing a weighted discrete bacterial optimization algorithm for feature gene selection, proving it could solve premature convergence issues in traditional bacterial optimization. Chen et al. [12] applied rough set and entropy calculation methods for selection, demonstrating effective improvement in tumor data classification accuracy. Wang et al. [13] introduced Markov blankets to improve Wrapper methods for gene feature selection, with results validating method effectiveness. In 2018, Wu Chenwen et al. [14] proposed a feature gene selection method based on ReliefF and ant colony optimization to address multi-classification problems in microarray data, achieving high multi-classification performance with fewer feature genes. Jain et al. [15] addressed gene classification and cancer diagnosis by integrating Correlation-based Feature Selection (CFS) and improved binary Particle Swarm Optimization (iBPSO) with Naive Bayes classifiers, attaining high classification accuracy.

These studies reveal that Filter-Wrapper combination methods have achieved

excellent results in gene data classification but still face two primary issues: (a) Most algorithms employ SVM classifiers, whose parameter selection significantly impacts classification results yet lacks unified standards or theoretical guidance; (b) Existing methods predominantly use single classifier models, with minimal research on ensemble learning approaches for gene feature classification, potentially limiting classification accuracy due to single classifier performance bottlenecks.

Kernel Extreme Learning Machine (KELM) demonstrates superior performance compared to SVM and BP Neural Networks (BPNN) [16]. Based on this analysis, to overcome these limitations and achieve higher classification accuracy, this paper proposes a novel classification model FICS-EKELM for high-dimensional gene expression data. The method first employs FCBF feature selection to eliminate redundant features and noise from datasets. It then uses bootstrap sampling for PCA transformation to generate multiple training subsets, upon each of which an Improved Crow Search Algorithm (ICS) simultaneously selects optimal feature subsets and optimizes KELM model parameters, producing diverse base KELM classifiers. Finally, an ensemble KELM classification model is constructed, with computational efficiency further improved through multi-threaded parallel implementation on multi-core platforms using OpenMP.

The innovations of this work are: (a) Applying FCBF method for feature dimensionality reduction in original high-dimensional gene expression data, eliminating redundant features and noise. Compared with other Filter methods, FCBF considers both inter-feature correlation and feature redundancy, demonstrating lower time complexity and better feature selection results in extensive experiments, while ReliefF, IG, and Fisher Score methods inadequately handle redundant features despite their ability to process incomplete and noisy data. (b) Proposing ICS algorithm for simultaneous feature subset selection and KELM parameter optimization to construct base classifiers. The Crow Search Algorithm (CSA) is simple to implement with few parameters, achieving comparable or even superior optimization results compared to Particle Swarm Optimization (PSO), Genetic Algorithm (GA), and Artificial Bee Colony (ABC). (c) Adopting ensemble classification thinking for gene feature selection and classification. (d) Implementing model parallel computation using OpenMP on multi-core processors to effectively improve algorithm efficiency.

## 1 Theoretical Background

### 1.1 FCBF Algorithm (Fast Correlation-Based Filter)

The Fast Correlation-Based Filter (FCBF) [17] is a typical heuristic sequential backward elimination method that uses symmetrical uncertainty measures to assess correlation between two features. The core concept employs Symmetrical Uncertainty (SU) as the metric standard: if a feature exhibits high uncertainty with class labels while showing low uncertainty with already-selected features, it is marked as an important feature.

The FCBF algorithm can be simply described as follows: Given a dataset $(x_i, t_i)$, $i = 1, \dots, N$, where $x_i = [x_{i1}, x_{i2}, \dots, x_{id}]^T \in \mathbb{R}^n$, $t_i = [t_{i1}, t_{i2}, \dots, t_{im}]^T \in \mathbb{R}^m$, and sample classes are $Y = (y_1, y_2, \dots y_N)$.

**Step 1**: Initialize $T$ and $S$; */T is the feature vector set, S is the feature subset/* **Step 2**: For each $t_i \in T$, calculate the SU value between features and classes, i.e., $SU(t_i, Y)$; **Step 3**: Select features from $T$ where $SU(t_i, Y) > r$, sort them in descending order by SU value, and store in $S'$; **Step 4**: Select a feature $t_i$ from $S'$, add $t_i$ to set $S$, and delete $t_i$ from $S'$; **Step 5**: Calculate the SU value $SU(t_i, t_j)$ between $t_i$ and $t_j$, removing redundant features of $t_j$. If $SU(t_i, t_j) > SU(t_i, Y)$, delete $t_i$ from $S'$; **Step 6**: Repeat Steps 4 and 5 until $S$ is empty; **Step 7**: Output the obtained feature subset $S$.

### 1.2 Kernel Extreme Learning Machine (KELM)

KELM, proposed by Huang et al. [18] based on the single-hidden-layer feedforward neural network model, can approximate any continuous target function with minimal error between its output values and class label values.

Assuming $N$ training sample sets $(x_i, t_i)$, $i = 1, \dots, N$, where $x_i = [x_{i1}, x_{i2}, \dots, x_{id}]^T \in \mathbb{R}^n$, $t_i = [t_{i1}, t_{i2}, \dots, t_{im}]^T \in \mathbb{R}^m$, with hidden layer activation function $g(x)$ and $L$ hidden layer nodes, the ELM output function is calculated as:

$$f_L(x) = \sum_{j=1}^{L} \beta_j h_j(x) = h(x)\beta$$

where $\beta_j = [\beta_{j1}, \beta_{j2}, \dots, \beta_{jL}]^T$ $(j = 1, 2, \dots, L)$ represents the output weight values connecting the $j$-th hidden layer node to output layer nodes. Here $H = \{h_{ij}\}$ $(i = 1, \dots, N; j = 1, \dots, L)$ is the hidden layer output matrix, where column $j$ of $H$ corresponds to the output vector of hidden node $j$ for inputs $x_1, x_2, \dots, x_n$, and row $i$ of $H$ corresponds to the output vector for input $x_i$. The output weight values for the linear system are typically determined using least squares:

$$\beta = H^+ T$$

where $H^+$ is the Moore-Penrose generalized inverse matrix of hidden layer output matrix $H$.

Subsequently, Huang et al. introduced kernel functions to avoid the problem of randomly generated input weights and bias values in ELM, proposing the kernel-based ELM method KELM. The KELM output weight calculation formula is:

$$\beta = (I/C + H^T H)^{-1} H^T T$$

Therefore, the KELM output function expression is:

$$f(x) = h(x)H^T(I/C + HH^T)^{-1}T$$

When the hidden layer mapping function $h(x)$ is unknown, the kernel function matrix is calculated as:

$$\Omega_{ELM} = HH^T : \Omega_{ELM_{ij}} = h(x_i) \cdot h(x_j) = K(x_i, x_j)$$

where $K(x_i, x_j)$ represents the kernel function. In KELM, the kernel function is the RBF kernel function:

$$K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2)$$

Thus, the KELM classification model output function expression is:

$$f(x) = \begin{bmatrix} K(x, x_1) \\ \vdots \\ K(x, x_N) \end{bmatrix}^T (I/C + \Omega_{ELM})^{-1}T$$

## 2 Improved Crow Search Algorithm (ICS)

The Crow Search Algorithm (CSA), proposed by Askarzadeh in 2016 [19], is a novel metaheuristic algorithm that simulates the intelligent foraging behavior of crows in nature. When solving optimization problems, $N$ crows are assumed to be randomly distributed in an $n$-dimensional search space, where $x_{i,t} = [x^1_{i,t}, x^2_{i,t}, \dots, x^n_{i,t}]$ $(i = 1, 2, \dots, N; t = 1, 2, \dots, \text{Maxiter})$ represents the position of crow $i$ at iteration $t$. $M_{i,t}$ denotes the memory value (optimal position) of crow $i$ at iteration $t$, $AP_{i,t}$ represents the awareness probability of crow $i$ at iteration $t$, and $fl_{i,t}$ denotes the flight length of crow $i$ at iteration $t$.

Traditional CSA initializes positions randomly:

$$x_{i,t} = x_{\min} + \text{rand} \cdot (x_{\max} - x_{\min})$$

where $x_{i,t}$ is the randomly generated position of crow, $x_{\max}$ and $x_{\min}$ are the maximum and minimum values of $x$, respectively, and rand is a random number uniformly distributed in $[0, 1]$.

However, random initialization cannot guarantee individual quality. Good initial solutions facilitate solution efficiency and quality, while poor initial solutions affect efficiency and increase uncertainty. An optimal initialization population ensures faster algorithm convergence. This paper employs chaotic maps to optimize crow search initialization. Chaotic motion is quasi-random behavior that

naturally emerges in deterministic nonlinear systems, possessing both deterministic processes and randomness [20]. Chaotic motion enables algorithms to escape local optima while seeking global optimal solutions. Therefore, this paper uses the Logistics chaotic mapping function to initialize crow positions:

$$X_{n+1} = \mu \cdot X_n \cdot (1 - X_n), \quad \mu \in [0, 4], X_0 \in (0, 1)$$

where parameter $\mu$ controls the degree of chaos.

Dynamic adjustment through awareness probability $AP$ achieves balance between global and local search. Since crow position updates affect optimal solutions and convergence speed, chaotic algorithms are introduced to further optimize position updates:

$$x_{i,t+1} = \begin{cases} x_{i,t} + r_i \cdot fl_{i,t} \cdot (m_{j,t} - x_{i,t}) \cdot w_i, & \text{if } r_j \geq AP_{i,t} \\ x_{i,t} + w_z \cdot (x_{\max} - x_{\min}) \cdot \text{rand}, & \text{otherwise} \end{cases}$$

where $w_i$ represents the chaotic mapping value obtained at generation $i$, $w_z$ represents the chaotic mapping value obtained at generation $z$, $AP_{i,t}$ denotes the awareness probability of crow $j$ at generation $t$, and $r_i$ and $r_j$ are random numbers uniformly distributed in $[0, 1]$.

As shown in equation (9), introducing hybrid functions further balances global and local search, enabling more flexible dynamic perturbation. In early stages, larger $w_i$ values ensure global search dominates, improving population diversity. In later stages, smaller $w_i$ values increase local search weight, accelerating convergence. When crow $i$ changes position, the memory update expression is:

$$M_{i,t+1} = \begin{cases} x_{i,t+1}, & \text{if } f(x_{i,t+1}) > f(M_{i,t}) \\ M_{i,t}, & \text{otherwise} \end{cases}$$

where $M_{i,t}$ represents the crow' s memory value and $f(M_{i,t})$ denotes the fitness value.

For binary crow search algorithms searching in discrete space where each solution is represented as 1 or 0, a mapping function $S(x)$ is introduced to transform continuous space values to discrete space $[0, 1]$:

$$S(x) = \frac{1}{1 + e^{-x}}$$

The mapping function expression is:

$$x_{i,t+1} = \begin{cases} 1, & \text{if } S(M_{i,t}) > \text{rand}() \\ 0, & \text{otherwise} \end{cases}$$

where rand() is a random number uniformly distributed in $[0, 1]$.

## 3 FICS-EKELM Model

This section details the FICS-EKELM model, whose overall architecture is shown in Figure 1 [Figure 1: see original paper].

### 3.1 Training Subset Generation

To ensure sample diversity, the rotation forest algorithm concept [21] is introduced. Bootstrap sampling randomly extracts samples from original data, followed by PCA transformation to generate new sample sets. Assuming original dataset samples are $X$ with class labels $Y$, and the new training set contains $k$ samples, the algorithm for generating training samples is as follows:

**Input**: Original datasets $X$
**Output**: sub_datasets $(T_1, T_2, \ldots, T_k)$

```
Begin
For i = 1 to k
    [sub_X, sub_Y] = randomsub (X);
    trainX_subnew = bootstrapal(sub_X, sub_Y);    /* perform sampling */
    Coeff = pcasky(trainX_subnew); /* perform PCA transformation to obtain new samples */
    New_sub (i) = trainX_subnew * R_coeff;
End For
Return: Final sub_datasets (T_1, T_2, ..., T_k)
```

### 3.2 Base Classifier Model Construction

Literature [22] explicitly states that achieving higher classification accuracy in ensemble classification models requires classifiers to be both accurate and diverse—ensemble models with greater diversity exhibit stronger performance. Therefore, constructing diverse classifiers is crucial. Similar to SVM, KELM is significantly influenced by its penalty factor $C$ and kernel width $\gamma$; improper values lead to poor classification performance. Constructing diverse KELM-based ensemble classifiers can be achieved through two conditions: (1) Bootstrap sampling and PCA feature transformation produce different training datasets, enabling each KELM model to receive different input samples and ensuring training on diverse datasets; (2) The penalty factor $C$ and kernel width $\gamma$, which critically affect KELM classification performance, are difficult to set manually. Using the ICS algorithm for optimization yields different classification models, thereby guaranteeing data diversity and classifier differences.

The core idea of the base classifier ICS-KELM is to utilize the ICA algorithm' s optimization mechanism to simultaneously perform feature subset selection and parameter optimization, thereby obtaining optimal base classifiers. The flowchart for constructing base classifier models is shown in Figure 2 [Figure 2: see original paper], with specific steps as follows:

a) Population initialization: Each individual in the population consists of multiple discrete feature attribute values and two continuous values for penalty factor $C$ and kernel width $\gamma$. The encoding form is $\tau = (1, 0, \ldots, 1, 1, C, \gamma)$, where 1 indicates selected features and 0 indicates unselected features.

b) Using parameters decoded from initialized individuals, KELM training is performed on training subsets to calculate each individual' s fitness value:

$$\text{Fitness} = \alpha \cdot \text{acc}_i + (1 - \alpha) \cdot \frac{N - |\text{Subset}|}{N}$$

where $\text{acc}_i$ represents the classification accuracy of solution $i$, $N$ is the total number of features, |Subset| denotes the number of features in the selected optimal subset, $\alpha$ is the weight balancing classification accuracy and subset size ($0 < \alpha < 1$, set to 0.8 in this paper), and Fitness represents the average K-fold Cross Validation (K-fold CV) value.

c) Increase iteration count;

d) Update population positions and memory values, comparing individual fitness values. If new fitness values exceed comparison values, update the optimal fitness value;

e) Train KELM using new positions and calculate fitness values according to equation (14);

f) If maximum population size is reached, proceed to step g; otherwise, return to step d;

g) Compare current fitness value with global optimal fitness value. If current value exceeds recorded optimal fitness, update to current value;

h) If maximum iteration count is reached, proceed to step i; otherwise, return to step c;

i) Output global optimal memory position as the optimal solution;

j) Train on training subsets using obtained optimal feature subsets and parameters to construct optimal base classifiers.

### 3.3 Ensemble Classifier Model

This paper employs weighted voting to integrate these different base classifiers into a unified model, with weight coefficients obtained through normalization of base classifier accuracy on validation sets. The combined classifiers produce final output results. For a given data sample $x$ with $K$ classifiers $T_k(x)$, $k = 1, 2, \ldots, K$, the majority voting strategy yields the final classification result:

$$y = \operatorname{sgn}\left(\sum_{k=1}^{K} \delta_k T_k(x)\right), \quad y \in \{-1, 1\}$$

where $\delta_{ij}$ represents classifier weights with $\delta_{ij} = 1$ if $i \neq j$ and $\delta_{ij} = 0$ otherwise. Equation (15) indicates that the final classification result corresponds to the class label with maximum cumulative output from $K$ classifiers $T_k(x)$.

### 3.4 Parallel Model Computation

For complex optimization problems, the ICS algorithm requires multiple updates to guarantee finding optimal solutions. ICS algorithm components including initial solution generation, fitness calculation, and population position updates are time-consuming but mutually independent, endowing the algorithm with natural parallelism. To fully exploit this parallelism and improve efficiency, this paper proposes implementing model parallel computation on multi-core processors using OpenMP [23]. The multi-core platform framework comprises three layers:

a) **Particle Layer**: Consists of a series of particles, with parallel algorithms controlling the entire ICS iteration process, where each particle independently participates in the complete computation.

b) **OpenMP Platform**: Ensures synchronization of parallel algorithms while establishing communication with the operating system. The platform' s core component is the scheduler, which provides job scheduling and allocation for the operating system.

c) **Multi-core Processor**: Jobs are called by the system through OpenMP at this layer.

The pseudocode for parallel model FICS-EKELM is as follows:

```
initialize model parameters
train KELM;
calculate the fitness;    /* fitness is the fitness value */
while t < max_iteration /* max_iteration is the maximum iteration count */
    for each solution
        update position;
        update memory;
        train KELM;
        calculate the fitness;
        calculate fitness_best;
        calculate memory_best;
    end for;
    calculate fitness_global;
    calculate memory_global;
    t = t + 1;
```

```
end while
```

## 4 Experiments

### 4.1 Experimental Setup

To evaluate the proposed method' s effectiveness on high-dimensional microarray gene expression data, six public high-dimensional gene datasets were selected: Breast Cancer, CNS, Leukemia, Lung Cancer, Lymphoma, and Prostate. Dataset information is shown in Table 1 .

**Table 1** Gene Dataset Information

| Dataset | Samples | Genes | Classes |
|---|---|---|---|
| Breast Cancer | 97 | 24481 | 2 |
| CNS | 60 | 7129 | 2 |
| Leukemia | 72 | 7129 | 2 |
| Lung Cancer | 181 | 12533 | 2 |
| Lymphoma | 77 | 7129 | 2 |
| Prostate | 102 | 12600 | 2 |

Experiments were conducted on a Windows 7 operating system with an Intel Core(TM) i5 processor at 3.2 GHz and 4 GB RAM, programmed in MATLAB 2014b. ELM and KELM utilized MATLAB toolboxes. ICS-KELM algorithm parameters are listed in Table 2 .

**Table 2** ICS-KELM Algorithm Parameter Settings

| ICS Algorithm Parameters | Value |
|---|---|
| Maximum Iterations | 100 |
| Flight Length fl | 2 |
| Awareness Probability AP | 0.1 |
| Population Size | 20 |

Additionally, ICS-EKELM model experimental results were compared with ELM, KELM, SVM, and BPNN methods. Detailed parameter settings: For fair comparison, KELM and SVM models used identical parameter settings with grid search method, where $C$ and $\gamma$ search ranges were $C \in \{2^{-11}, \ldots, 2^{11}\}$ and $\gamma \in \{2^{-11}, \ldots, 2^{11}\}$. Hidden layer node counts for ELM and BPNN were obtained through trial-and-error methods, with 18 and 21 hidden nodes respectively.

### 4.2 Experimental Results and Discussion

To validate the proposed method' s effectiveness, experiments first present classification results on six datasets compared with four common methods based on ReliefF, mRMR, IG, and CFS feature selection, as shown in Table 3 . The table presents classification accuracy and standard deviation values for each algorithm. Results demonstrate that the proposed method achieves the highest classification accuracy among the five models, with ReliefF, mRMR, IG, and CFS-based methods showing significantly lower performance. For example, on the Breast Cancer dataset, the proposed method achieved 92.98% average classification accuracy, while other methods obtained only 88.42%, 85.57%, 83.51%, and 81.92% respectively. These results also confirm that feature selection combined with ensemble classification learning effectively improves high-dimensional gene data classification accuracy. Furthermore, the small standard deviation values of the proposed method demonstrate its good stability.

**Table 3** Comparison of Classification Accuracy Among Five Algorithms

| Dataset | Proposed Method | ReliefF | Fisher Score | IG | mRMR |
|---|---|---|---|---|---|
| Breast Cancer | 92.98 ± 0.21 | 88.42 ± 0.30 | 85.57 ± 0.34 | 83.51 ± 0.42 | 81.92 ± 0.33 |
| CNS | 91.87 ± 0.27 | 87.44 ± 0.44 | 84.42 ± 0.42 | 83.44 ± 0.44 | 82.73 ± 0.73 |
| Leukemia | 99.71 ± 0.18 | 96.33 ± 0.33 | 94.38 ± 0.25 | 95.45 ± 0.45 | 95.25 ± 0.31 |
| Lung Cancer | 90.79 ± 0.29 | 86.54 ± 0.54 | 84.38 ± 0.39 | 85.44 ± 0.44 | 83.47 ± 0.36 |
| Lymphoma | 100.00 ± 0.25 | 96.31 ± 0.39 | 94.47 ± 0.30 | 95.36 ± 0.47 | 94.30 ± 0.30 |
| Prostate | 97.43 ± 0.31 | 93.39 ± 0.39 | 91.47 ± 0.36 | 92.30 ± 0.30 | 91.43 ± 0.30 |

To fully demonstrate the effectiveness of feature selection, Table 4 presents the number of features selected by five feature selection algorithms across six datasets. The proposed method selects the fewest features, followed by FCS and ReliefF. This occurs because the proposed method first uses FCBF to filter out numerous irrelevant and noisy features, then employs ICS search to further optimize feature subset selection, effectively eliminating highly redundant features.

**Table 4** Comparison of Feature Counts Selected by Five Methods

| Dataset | Proposed Method | ReliefF | IG | mRMR | CFS |
|---|---|---|---|---|---|
| Breast Cancer | 8 | 12 | 15 | 18 | 21 |
| CNS | 9 | 13 | 16 | 19 | 22 |
| Leukemia | 4 | 8 | 11 | 14 | 17 |
| Lung Cancer | 9 | 14 | 17 | 20 | 23 |
| Lymphoma | 7 | 11 | 14 | 17 | 20 |
| Prostate | 5 | 9 | 12 | 15 | 18 |

For comprehensive performance evaluation, Table 5 compares the proposed method with commonly used classification models: SVM, ELM, BPNN, and Naive Bayes. Results show the proposed method significantly outperforms these four methods due to simultaneous feature selection and model parameter optimization using ICS, plus ensemble classification construction that overcomes overfitting and classification bottlenecks in single classifiers, further improving accuracy.

**Table 5** Comparison of Classification Results Among Five Methods Based on Different Classifiers

| Dataset | Proposed Method | SVM | ELM | BPNN | Naive Bayes |
|---|---|---|---|---|---|
| Breast Cancer | 92.98 | 87.44 | 85.57 | 83.51 | 81.92 |
| CNS | 91.87 | 84.42 | 83.44 | 82.73 | 80.33 |
| Leukemia | 99.71 | 94.38 | 95.45 | 95.25 | 93.47 |
| Lung Cancer | 90.79 | 84.38 | 85.44 | 83.47 | 81.36 |
| Lymphoma | 100.00 | 94.47 | 95.36 | 94.30 | 92.43 |
| Prostate | 97.43 | 91.47 | 92.30 | 91.43 | 89.57 |

The number of base classifiers affects results. Since unified theoretical guidance for ensemble classifier quantity remains unavailable, selection typically relies on experimental trial. This experiment set ensemble classifier quantity range to $[1, 10]$, selecting the parameter value corresponding to optimal classification results for subsequent experiments, as shown in Figure 3 [Figure 3: see original paper]. The figure reveals that classification accuracy improves significantly as classifier quantity increases from 1. Peak accuracy occurs at 5 classifiers, after which performance does not improve with further increases but shows fluctuation, indicating that beyond a certain threshold, additional classifiers do not enhance classification performance.

To validate ICS algorithm' s global search capability and convergence speed, experiments further investigate the iteration mechanism. Using Lung Cancer dataset as an example, Figure 5 [Figure 5: see original paper] shows optimal fitness value evolution during 5-fold CV (first fold) for ICS and original CSA

algorithms. The figure tracks global optimal value changes, recording the best fitness value among all individuals at each iteration. Analysis shows ICS curve performs better, evolving rapidly from first iteration and converging to maximum value by the 23rd iteration, after which it stabilizes. The CSA curve converges to a lower value by the 19th iteration, remaining stable but below ICS curve, suggesting CSA may fall into local optima without finding global optimum. This demonstrates ICS possesses superior global search capability and convergence speed, rapidly converging to global optimal solutions.

To verify parallel model performance, parallel and serial models were compared. Table 6 presents training time and classification accuracy comparisons on six datasets. Both models show very similar classification accuracy results, with minor differences arising from random dataset selection during cross-validation. However, serial model actual time consumption is significantly higher than parallel model. Figure 4 [Figure 4: see original paper] compares running times for parallel and serial models on CNS dataset across 5-fold CV independent runs. Serial model CPU average computation time is approximately 2.6 times that of parallel model PHGSA-KELM, with parallel model spending far less time in each fold. This indicates the proposed method benefits from parallel algorithms, compensating for excessive time consumption in serial algorithm iterative optimization and improving computational efficiency.

**Table 6** Comparison of Training Time and Classification Accuracy Between Parallel and Serial Models

| Dataset | Parallel Model | Serial Model |
|---|---|---|
| | Training Time (s) | Accuracy (%) |
| Breast Cancer | 125.3 | 92.98 |
| CNS | 118.7 | 91.87 |
| Leukemia | 95.4 | 99.71 |
| Lung Cancer | 142.6 | 90.79 |
| Lymphoma | 89.3 | 100.00 |
| Prostate | 108.5 | 97.43 |

Results from Figures 3-5 and Tables 3-6 demonstrate the proposed method achieves average classification accuracies of 92.98%, 91.87%, 99.71%, 90.79%, 100.00%, and 97.43% on six public gene datasets, while significantly reducing feature counts to 8, 9, 4, 9, 7, and 5 respectively from original high-dimensional feature spaces. The minimal classification accuracy degradation validates feature selection effectiveness, attributable to the evaluation function design that simultaneously considers feature selection and classifier performance, minimizing feature quantity while maximizing classification results. Parallel computation saves nearly 2/3 of CPU average computation time compared to serial computation, proving the approach fully exploits ICS parallelism and improves algorithm efficiency.

## 5 Conclusion

To better address high-dimensional gene data classification challenges, this paper fully leverages FCBF filtering performance, crow search capability, and ensemble classification model advantages to propose a classification model based on FCBF and ensemble optimized KELM classifiers. In this method, FCBF effectively removes redundant features and noise from datasets, identifies feature sets highly relevant to classification results, and reduces feature dimensionality. Simultaneously, adopting ensemble classification thinking, KELM serves as base classifiers while ICS algorithm simultaneously achieves optimal feature subset selection and KELM model parameter optimization to obtain optimal classification models. Multi-core processor implementation using OpenMP achieves model parallel computation, further enhancing algorithm efficiency. Experimental results demonstrate the proposed method outperforms typical filter feature selection methods and classification methods based on SVM, ELM, BPNN, and Naive Bayes. The method not only removes redundant genes but also achieves high classification performance, with experiments validating its effectiveness and validity.

Future work will investigate feature and feature subset relationship metrics. Additionally, flexible adjustment of ICS algorithm parameters to reduce parameter setting impacts on search and classification results, enabling further algorithm optimization, represents worthwhile research.

## References

[1] Boulesteix A L, Strobl C, Augustin T, et al. Evaluating microarray based classifiers: an overview [J]. Cancer Informatics, 2008, 6: 77–97.

[2] Lin Hungyi. Reduced gene subset selection based on discrimination power [J]. Neurocomputing, 2017, 260: 313-320.

[3] Hanaa S, Gamal A, Nawal E. Classification of human cancer diseases by gene expression profiles [J]. Applied Soft Computing, 2017, 50: 124-134.

[4] Liang Sen, Ma Anjun, Yang Sen, et al. a review of matched pairs feature selection methods for gene expression data analysis [J]. Computational and Structural Biotechnology, 2018, 16: 88-97.

[5] Ghaddar B, Naoum-Sawaya J. High dimensional data classification and feature selection using support vector machines [J]. European Journal of Operational Research, 2018, 265 (3): 993-1004.

[6] Bolón-Canedo V, Sanchez-Marono N, Alonso-Betanzos A. A review of feature selection methods on synthetic data [J]. Knowledge and Information Systems, 2013, 34 (3): 483−519.

[7] Jerzy K, Tomasz L. The feature selection bias problem in relation to high-dimensional gene data [J]. Artificial Intelligence in Medicine, 2016, 66: 63-71.

[8] 谢娟英, 胡秋锋, 董亚非. K-S 检验与 mRMR 相结合的基因选择算法 [J]. 计算机应用研究, 2016, 33 (4): 1013-1018. (Juanying Xie, Qiufeng Hu, Yafei Dong. Gene selection algorithm based on K-S test and mRMR [J]. Application Research of Computers, 2016, 33 (4): 1013-1018.)

[9] Lai C M, Yeh W C, Chang C Y. Gene selection using information gain and improved simplified swarm optimization [J]. Neurocomputing, 2016, 218: 303-312.

[10] Lu Huijuan, Chen Junying, Yan Ke, et al. A hybrid feature selection algorithm for gene expression data classification [J]. Neurocomputing, 2017, 256: 60-68.

[11] Wang Hong, Jing Xingjian, Niu Ben. A discrete bacterial algorithm for feature selection in classification of microarray gene expression cancer data [J]. Knowledge-Based Systems, 2017, 126: 8-19.

[12] Chen Yumin, Zhang Zunjun, Zheng Jianzhong, et al. Gene selection for tumor classification using neighborhood rough sets and entropy measures [J]. Journal of Biomedical Informatics, 2017, 67: 59-68.

[13] Wang Aiguo, An Ning, Yang Jing, et al. Wrapper-based gene selection with Markov blanket [J]. Computers in Biology and Medicine, 2017, 81: 11-23.

[14] 吴辰文, 李晨阳, 郭叔瑾, 等. 基于 ReliefF 和蚁群算法的特征基因选择方法 [J]. 计算机应用研究, 2018, 35 (9): 1-7. (Wu Chenwen, Li Chenyang, Guo Shujin, et al. Feature gene selection method based on ReliefF and ant colony optimization [J]. Application Research of Computers, 2018, 35 (9): 1-7.)

[15] Jain I, Vinod K, Jain R. Correlation feature selection based improved binary particle swarm optimization for gene selection and cancer classification [J]. Applied Soft Computing, 2018, 62: 203-215.

[16] Luo Fangfang, Guo Wenzhong, Yu Yuanlong. A multi-label classification algorithm based on kernel extreme learning machine boosting for molecular classification [J]. Knowledge-Based Systems, 2018, 142: 181-191.

[17] Song Q B, Ni J J, Wang G T. A fast clustering-based feature subset selection algorithm for high-dimensional data [J]. IEEE Trans on Knowledge and Data Engineering, 2013, 25 (1): 1-14.

[18] Huang Guangbin. Extreme learning machine for regression and multiclass classification [J]. IEEE Trans on Systems, Man, and Cybernetics, Part B: Cybernetics, 2012, 42 (2): 513-529.

[19] Askarzadeh A. A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm [J]. Computers & Structures, 2016, 169: 1–12.

[20] Wang G G, Guo L, Gandomi A H, et al. Chaotic krill herd algorithm [J]. Information Sciences, 2014, 274: 17–34.

[21] Rodriguez J, Kuncheva L, Alonso C J. Rotation forest: a new classifier ensemble method [J]. IEEE Trans on Pattern Analysis and Machine Intelligence, 2006, 28 (10): 1619-1630.

[22] Hansen L K, Salamon P. Neural network ensembles [J]. IEEE Trans on Pattern Analysis and Machine Intelligence, 1990, 12 (10): 993-1001.

[23] Chapman B, Jost G, Van R. Using OpenMP: portable shared memory parallel programming [M]. Cambridge: MIT Press, 2007.

*Note: Figure translations are in progress. See original paper for figures.*