Chapter 3: AN IMPROVED BINARY PSO FOR GENE SELECTION USING DNA MICROARRAY GENE EXPRESSION DATA

3.1 Introduction

DNA microarray experiments make possible to study the expression levels of thousands of genes simultaneously that take part in important genetic processes within a cell and across collections of related samples. These experiments provide molecular insights into the mechanisms of living organisms by measuring the activities and interaction of thousands of genes concurrently. One of the major applications of microarrays is identification of highly relevant genes which is responsible for a specific disease such as cancer. An early and accurate prognosis of cancer facilitates the proper line of treatment, and DNA microarray technology has shown great potential in diagnosis of cancer and its classification. However, the huge amount of data produced by microarrays result in a great challenge in its analysis using data mining techniques. Specially, in classification task the outcomes are adversely affected by the high dimensionality of the microarray data that also include irrelevant, redundant and noisy genes. In addition, only a small number of tissue samples relative to the high dimensional gene expressions, are available for the analytical study lead to high misclassification errors. Thus, selection of most relevant genes prior to classification is the key aspect in the design of any cancer classification model. Generally a cancer classification model has two steps; (1) identification of most relevant genes (2) prediction of type or subtype of an unknown tissue sample. An efficient gene selection method is required in order to achieve better classification accuracies using low dimensional gene subset that reduces computational costs.

In this chapter, we propose an improved Binary Particle Swarm Optimization (iBPSO) based wrapper approach for gene selection using microarray gene expression data. We have applied three induction algorithms in this wrapper approach namely, Naive-Bayes (NB), k-Nearest Neighbor (k-NN) and Support Vector Machine (SVM) to evaluate the classification
performance of each gene set using five-fold cross-validation strategy.

3.2 Binary Particle Swarm Optimization

Particle Swarm Optimization is a swarm intelligence based Evolutionary Computation (EC) technique [92, 93]. It finds a solution to an optimization problem by simulation of the social behavior of bird flocks and fish schools. It is a global search technique that searches a problem feature space for an optimal solution in successive iterations by managing a swarm of particles (subset of genes). Each particle (possible gene subset) maintains its current and best so far position ($p_{best}$) in every iteration and a global best position ($g_{best}$) attained by the swarm is also updated across iterations. Each particle evaluates its current position using a predefined fitness function and then move through the feature space with a velocity determined by positions and processed fitness values.

Standard BPSO (Binary Particle Swarm Optimization) is a discrete version of PSO which is originally introduced in 1997 by Kenndey and Eberhart [23]. In BPSO, a particle is represented as a string of binary digits 0 or 1. The position of the particle is updated by flipping each bit value between 0 or 1 based on the velocity of that bit. In a binary space, a velocity of a particle can be defined as changes of probabilities that a bit flips its current state or not. A particle’s movements may be considered from one corner to other corners of a hypercube by flipping various numbers of bits; thus, the overall particle velocity may be described by the Hamming distance(number of bits changed) between particle in two consecutive iterations.

However, the problem of moving around a local optimum is very common in BPSO like other Evolutionary Computation (EC) techniques, which hinders the algorithm to obtain an optimal solution globally. The very characteristic of microarray data is high dimensionality (very large number of genes) and small number of tissue samples, which makes it easy to be trapped in a local optimum. A particle updates its position by using two fitness values i.e. $p_{best}$ and $g_{best}$ in each iteration. If global fitness value $g_{best}$ is stuck in a local optimum then it diminishes the exploration capability of the whole swarm. Thereby the algorithm outputs a false optimum and a superior classification accuracy can not be achieved. Thus, we present a solution for the problem of early convergence at a local optimum in an improved Binary Particle Swarm
3.3 System Model

DNA microarray experiments measure thousands of gene expressions for tissue samples and these data are stored in the form of microarray data matrix. It is well known that variations in the systematic patterns of gene expressions exhibited by a specific cell type is correlated with the biological variations of a particular cancer type [3]. Let $M$ genes form $M$-dimensional gene expression space $E^M$ corresponding to a sample space $U$ then this association can be represented mathematically as

$$f : E^M \rightarrow U$$

More specifically, there are $l$ target classes for all the given samples then there exist an association between gene expression patterns and the $i^{th}$ class which can be defined as
\[ C^{(i)} = f(S^M_j) \]

where \( C^{(i)} \in U \) for \( 1 \leq i \leq l \) and \( S^M_j \in E^M \). Typically the gene expression vector \( S^M_j \) is an ordered sequence tuple with respect to the \( M \) genes for the \( j^{th} \) \( (1 \leq j \leq N) \) sample

\[ S^M_j = (e_1, e_2, e_3, \ldots, e_M) \]

in which each gene expression value denotes a feature. Suppose the whole microarray dataset \( D \) consists of \( N \) observations (samples) which can be represented as

\[ D = \{(S^M_j, C^{(i)}) : j = 1, 2, \ldots, N, i \in (1, 2, \ldots, l)\} \subseteq E^M \times U \]

The vector \( S^M_j (j = 1, 2, \ldots, N) \) consists of gene expression values for \( M \) number of genes and \( C^{(i)} \in \{1, 2, 3, \ldots, l\} \) is the class label assigned to the gene expression vector.

The main aim of this task is to develop a classification model which can predict the class labels for the given samples on the basis of a low-dimensional set of most relevant genes.

### 3.4 Proposed Methodology

#### 3.4.1 Improved Binary Particle Swarm Optimization (iBPSO)

Particle Swarm optimization (PSO) is a randomized wrapper approach based on swarm intelligence, introduced by Kennedy and Eberhart [92, 93, 23]. Binary PSO (BPSO) is a discrete version of original PSO. A swarm of particles move through a feature space to obtain an optimal solution for a given objective function in a limited number of iterations. Each particle’s state in a iteration is defined by two values first its position in the search space and second the velocity value to modify the position. In each iteration, every particle (or a candidate solution) updates its current position and velocity based on the best positions obtained previously by the particle \( (p_{best}) \) and by the whole swarm \( (g_{best}) \).

Let the swarm \( (\Omega) \) have \( W \) particles, where each particle represents a gene subset of \( m \) di-
3.4: iBPSO Chapter 3: iBPSO gene selection method

dimensions and $T$ is the total number of iterations. The particle in the swarm is defined as a string of $m$ binary values where a bit value 1 denotes the presence of that gene whereas a bit value 0 denotes absence of the gene in the subset. The position and velocity values of $k^{th}$ particle (i.e., gene subset) at $t^{th}$ iteration is defined as

$$P^t_k = (p^t_{k1}, p^t_{k2}, \ldots, p^t_{km}) \quad k = 1, 2, \ldots, W; \quad t = 1, 2, \ldots, T$$

$$V^t_k = (v^t_{k1}, v^t_{k2}, \ldots, v^t_{km}) \quad k = 1, 2, \ldots, W; \quad t = 1, 2, \ldots, T$$

The process starts with an initial population of particles say $\Omega_0$ chosen randomly, then initial position and velocity of a particle in the population is given by

$$P^0_k = (p^0_{k1}, p^0_{k2}, \ldots, p^0_{km}) \quad k = 1, 2, \ldots, W$$

$$V^0_k = (v^0_{k1}, v^0_{k2}, \ldots, v^0_{km}) \quad k = 1, 2, \ldots, W$$

A new population $\Omega_{t+1}$ is produced from a present population $\Omega_t$ by flipping the bits of the positions of particles in the present population by using Eq. 3.1 and Eq. 3.2. This modification is regulated by velocity value (i.e., probability measure) $v_{kd} (d \in m)$ which, in turn, is used to calculate the possible value of a dimension in position vector $p_{kd}$ as a 0 or 1, using following formula:

$$v_{kd}^{t+1} = w^t \times v_{kd}^t + c_1 \times \psi_1^t \times (p^t_{best,k} - p^t_{kd}) + c_2 \times \psi_2^t \times (g^t_{best} - p^t_{kd}), \quad (3.1)$$

finally a logistic (sigmoid) function is used to define the change in the bit position of each particle in swarm as

$$p_{kd}^{t+1} = \begin{cases} 1 & \text{if} (S\text{ig}(v_{kd}^{t+1}) > \psi_3^t) \\ 0 & \text{otherwise} \end{cases} \quad (3.2)$$

where, $S\text{ig}(v_{kd}^{t+1}) = 1/(1 + e^{-v_{kd}^{t+1}})$. $w^t$ is the inertial weight component of the velocity value which controls the exploration of the search space in the current iteration. Here the weight component $w^t$ is decreased linearly in each iteration. Thus the value of $w^t$ changes dynamically in each iteration and regulates the velocity of all particles. An initially large value
of $w^t$, provides better exploration of the search space to all particles in swarm and gradually decreasing value facilitates faster convergence (better exploitation) to the global solution. The following equation is used to compute the value of $w^t$ in each iteration $t$:

$$w^t = w_{max} - \frac{(w_{max} - w_{min}) \times (t - 1)}{T - 1}$$  \hspace{1cm} (3.3)

here, $w_{max}$ and $w_{min}$ are the initial and final values of the inertial weight component respectively, $t$ is the current iteration and $T$ is total number of iterations. $v_{kd}^t$, $v_{kd}^{t+1}$ and $p_{kd}^t$, $p_{kd}^{t+1}$ are the velocity and position vectors of $d^{th}$ dimension for particle $k$ in $t^{th}$ and $(t+1)^{th}$ iterations respectively, where $v_{kd}$ are limited in the range $[v_{min}, v_{max}]$ in our case it is [-6,6]. $c_1$ and $c_2$ are positive acceleration constants which regulate the effect of particle’s best and global best positions respectively; $\psi_1^t$, $\psi_2^t$ and $\psi_3^t$ are positive random values generated from a uniform distribution $U(0.0,1.0)$ in $t^{th}$ iteration. $p_{best,k}^t$ is the personal best position of particle $k$ refers to the position (i.e., gene subset) in the gene search space, where particle had the smallest misclassification error ($E$) as determined by five-fold cross validation, found in process till iteration $t$. Whereas $g_{best,d}^t$ is the global best position refers to the position (gene subset) having minimum misclassification error ($E$) amongst all the $p_{best,k}^t$ found by the swarm till iteration $t$.

The personal best position $p_{best,k}^t$ of $k^{th}$ particle can be updated for $t + 1$ iteration as follows:

$$p_{best,k}^{t+1} = \begin{cases} p_{best,k}^t & \text{if}(E(D, p_{kd}^{t+1}) \geq E(p_{best,k}^t)) \\ p_{kd}^{t+1} & \text{otherwise} \end{cases}$$  \hspace{1cm} (3.4)

and the global best position $g_{best}^{t+1}$ for $(t + 1)^{th}$ iteration is calculated as follows:

Let $p_{best,min}^t$ is the position vector of the particle which has lowest $E$ in $t^{th}$ iteration and $E_{min}^t$ is the minimum value of $E$ obtained in $t^{th}$ iteration. Then

$$g_{best}^{t+1} = \begin{cases} g_{best}^t & \text{if}(E_{min}^t \geq E(D, g_{best}^t)) \\ p_{best,min}^t & \text{otherwise} \end{cases}$$  \hspace{1cm} (3.5)

Note that when two particles are found having same misclassification error $E$ in $t^{th}$ iteration, the one with smaller number of genes is selected by the algorithm.
Like other Evolutionary Computation (EC) techniques, standard BPSO also suffers from a common limitation that it converges to a local optimum prematurely which prevents the swarm from getting a global solution. Thus the $g_{best}^t$ position shows no improvement over several generations of population. To overcome this problem, in iBPSO we proposed a rule to update the $g_{best}^t$ position in a manner so that all particles get a new direction to move towards the global best position. After a fixed number of generations if current $g_{best}^t$ position retains the same value, then we update every dimension of it using following rule:

$$g_{best,d}^t = \begin{cases} 
1 & \text{if} (P(p_{best,kd}^t = 1) \geq (P(p_{best,kd}^t = 0)) \\
0 & \text{otherwise}
\end{cases} \tag{3.6}$$

where $k = 1, 2, \ldots, W$ and $d = 1, 2, \ldots, m$.

The previous value of $g_{best}^t$ is updated by this new $g_{best}^t$, such that all particles come out of a local stagnation, and explore the gene search space for a low dimensional gene subset and achieves a minimum misclassification error ($E$). The proposed improved BPSO attains minimum misclassification error for an effectively reduced gene subset $g_i$.

To evaluate the discriminative efficiency of selected genes, first of all, genes present in the $k^{th}$ particle, i.e. where $p_{kd} = 1$, are used to find a gene subset $g_i$ which reduces the gene dimensions to $m$ in the final dataset $D$. The dataset $D$ contains $N$ gene expression vectors $s_j \in S_j$, $j = 1, 2, \ldots, N$ each of dimensionality $m$. The performance of each particle (gene subset) of every newly generated population is evaluated using three classifiers namely, Naive-Bayes (NB), 1-NN and, SVM. In the present study, these algorithms are chosen as the classifiers since these methods are most commonly used on two-class or multi-class microarray classification problems [94, 95, 96, 97, 87, 88]. Here, we have provided the brief overview of these algorithms.

**Naive-Bayes (NB)** [95, 98, 99] is a probabilistic learning approach, which works on an underlying assumption that all the feature values are conditionally independent given a corresponding target class. Then following equation is used by the NB classifier to predict the target
class of the given gene expression values \((e_1, e_2, \ldots, e_m)\) for the selected gene subset \((g_i)\) that defines the sample vector \(s_j\):

\[
c^{(i)}_{\text{NB}} = \arg\max_{c^{(i)} \in C^{(i)}} P(c^{(i)}) \prod_{j=1}^{m'} P(e_j|c^{(i)})
\]

where \(c^{(i)}_{\text{NB}}\) is the predicted class value for test sample which have maximum probability assigned by NB classifier. The probabilities \(P(c^{(i)})\) and \(P(e_j|c^{(i)})\) are simply estimated by counting the frequencies over the training data. NB does not need any exhaustive search of the given feature space. This hypothesis makes it simple and computationally fast classifier and has shown comparable performance in this domain.

**K-Nearest Neighbor (k-NN)** is an instance-based learning method introduced by Fix and Hodges in 1951 [100, 101]. In this method all instances are referred to as points in Euclidean space. This method simply stores the presented training data and constructs a local approximation to the target function that applies in the neighborhood of the new query instance to determine its target class. The k-NN finds k nearest neighbors of a query instance by calculating its Euclidean distance with each training instances and assigns most common value of target class among k nearest training examples to the query instance. In 1-NN, the predicted class of a query instance is the class of its nearest training instance. The k-NN method is simple and models complex target functions by a collection of less complex local approximations.

The main difficulties include high cost of classifying a new instance each time, determining the appropriate value of k, and the negative impact of irrelevant features on the distance metric. To reduce the computational cost, use of some indexing techniques can be a significant practical issue. Here, we have chosen 1-NN for the classification task.

**Support Vector Machines (SVM)** [102, 103, 104, 105] have been introduced for solving pattern recognition tasks and also known as large margin classifier. In SVM, all instances are represented as points in n-dimensional feature space and a decision boundary (hyperplane) is determined to separate the classes with maximum possible margin on either side. While constructing such optimal hyperplane only a small amount of the training data, the support vectors, have been considered which determine this margin [103]. Support vectors are the small subset of training data points which make the basis of the target function. These examples are
nearest to the decision boundary and lie on the margin. The existence of such support vectors
decide the position of the decision boundary. Let a training set of \( n \) samples \( \{y_j, x_j\}, j = 1, 2, ..., n \) is given, where \( x_j (\in S_j, j = 1, 2, ..., N) \) is the \( j^{th} \) sample point and \( y_j \in U \) is the \( j^{th} \) class label, a classifier of the form is defined using the support vector method approach:

\[
y(x) = \text{sign} \left[ \sum_{j=1}^{n} \alpha_j y_j \kappa(x, x_j) + b \right]
\]

where \( \alpha_j \) are positive real constants and \( b \) is a real constant. \( \kappa(\cdot, \cdot) \) is a kernel function which can be a linear function, a polynomial function or a RBF (Radial basis function). \( x_j \) is a training sample and \( x \) is a sample to be classified. For a binary classification such as \( y_j \in (+1, -1) \), the classifier is constructed as follows by assuming:

\[
w^T \psi(x_j) + b \geq 1, \quad \text{if} \quad y_j = +1,
\]

\[
w^T \psi(x_j) + b \leq -1, \quad \text{if} \quad y_j = -1
\]

which is equivalent to

\[
y_j[w^T \psi(x_j) + b] \geq 1, \quad j = 1, ..., n
\]

where \( \psi(\cdot) \) is a nonlinear function which maps the input space into a higher dimensional space. In this study, we have used RBF kernel function. We have applied a one-versus-rest (OVR) strategy for multi-class datasets, since it has shown superior performance on microarray classification to other strategies [106].

### 3.4.2 Classification Accuracy

The accuracy estimation of the proposed method is done for each dataset using five-fold cross validation misclassification error (\( \mathcal{E} \)). We have partitioned the dataset \( D \) into 5 disjoint sets \( (D_1, D_2, \ldots, D_5) \) of similar size using 5-fold cross-validation. One of the three classifiers learned on the basis of \( (D - D_v) \) training data and then this learned hypothesis is used to classify the samples in the set \( D_v \) by applying the corresponding methods of a classifier. This process is repeated five times so that each set \( D_v \) is used once as the test set. Finally a misclassification error for 5-fold cross validation (\( \mathcal{E} \)) is calculated as follows:

\[
\mathcal{E}(D, \mathbf{g}_i) = \frac{1}{5} \sum_{v=1}^{5} \mathcal{E}_v(D_v, \mathbf{g}_i) \quad (3.8)
\]
3.4: iBPSO

Chapter 3: iBPSO gene selection method

where,

\[ E_v(D_v, g_i) = \left| \left\{ (x_j^m, c^{(i)}) \in D_v : c_e^{(i)} \neq c^{(i)} \right\} \right| / |D_v| \]  \hspace{1cm} (3.9)

\( c_e^{(i)} \) is the predicted class of a query sample by the classifier.

3.4.3 Solution Algorithm

In this chapter, we have proposed an improved Binary Particle Swarm Optimization (iBPSO) method for gene selection and cancer classification using microarray data. Here three classifiers namely, Naive-Bayes, 1-NN and, SVM with 5-fold cross-validation are used to evaluate the classification performance of the proposed method. The detailed solution algorithm for iBPSO has been summarized as follows:

Step 1: Preprocess the dataset \( \mathcal{D} = \{(S_j^M, C(i)) : j = 1, 2, \ldots, N, i \in (1, 2, \ldots, l)\} \) to get 500 top-ranked genes.

Step 2: Set the parameters for iBPSO and generate an initial swarm \( \Omega \) of \( W \) particles randomly, where each particle have an initial position \( P^0_k = (p^0_{k1}, p^0_{k2}, \ldots, p^0_{km}) \) and velocity \( V^0_k = (v^0_{k1}, v^0_{k2}, \ldots, v^0_{km}) \) \( k = 1, 2, \ldots, W \).

Step 3: Adjust \( w' \) inertial weight component with Eq. 3.3

Step 4: For each particle in the swarm, calculate the fitness using cross-validated misclassification error \( (E(D, g_i)) \) for training set \( D \) and gene subset \( g_i \) using the classifier with 5-fold cross-validation.

Step 5: Update each particle’s personal best position \( p^t_{\text{best}, k} \) of particle \( k \) by using Eq. 3.4 and the global best position \( g^t_{\text{best}} \) according to Eq. 3.5 using the fitness evaluation results. The number of selected genes is also considered.

Step 6: If \( g^t_{\text{best}} \) does not change for \( k \) iterations, go to next step. Otherwise go to step 8.

Step 7: Replace current \( g^t_{\text{best}} \) according to Eq. 3.6.

Step 8: Update the velocity and position vector of each particle according to Eq. 3.1 and Eq. 3.2 respectively.

Step 9: Repeat Steps 4 to 8 until convergence or a certain number of iterations has been completed. Consequently, the best gene subset is obtained.
3.5 Experimental Setup and Results

3.5.1 Gene expression data sets and Parameter settings

For experimentation six benchmark microarray cancer datasets, which are obtained from http://csse.szu.edu.cn/staff/zhuzx/Datasets.html, are used to evaluate the performance of the proposed iBPSO method. Description of these microarray datasets are given in Table 3.1, includes number of observed tissue samples, number of genes per sample and number of classes. The datasets used in the experiments have large number of dimensions (thousands of genes) and consist of two and more than two classes, which are
3.5 Experimental Setup and Results

Chapter 3: iBPSO gene selection method

Table 3.1: Descriptive summary of Microarray datasets

<table>
<thead>
<tr>
<th>Datasets</th>
<th>No. of Total Genes</th>
<th>No. of Samples</th>
<th>No. of Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Central Nervous System (CNS)</td>
<td>7129</td>
<td>60</td>
<td>2</td>
</tr>
<tr>
<td>ALL-AML</td>
<td>7129</td>
<td>72</td>
<td>2</td>
</tr>
<tr>
<td>Breast</td>
<td>24,481</td>
<td>97</td>
<td>2</td>
</tr>
<tr>
<td>Lymphoma</td>
<td>4026</td>
<td>62</td>
<td>3</td>
</tr>
<tr>
<td>MLL</td>
<td>12,582</td>
<td>72</td>
<td>3</td>
</tr>
<tr>
<td>SRBCT</td>
<td>2308</td>
<td>83</td>
<td>4</td>
</tr>
</tbody>
</table>

appropriate to show the effectiveness of our approach. Thus these datasets represent binary and multi-class cancer classification problems. The cancer data sets are of Leukemia 2-class (ALL-AML), Central Nervous System (CNS), Breast, Lymphoma, Mixed-lineage leukemia (MLL) and, Small round blue cell tumors (SRBCT).

Firstly, we replaced the missing values of gene expressions with mean values and all the datasets are standardized to have mean equal to zero and standard deviation equal to one using following equation:

$$e_{new} = \frac{e - \mu}{\sigma}$$  \hspace{1cm} (3.10)

Here $\mu$ is the mean value and $\sigma$ is the standard deviation for the given vector. $e_{new}$ is the transformed value for a gene expression $e$. Then we pre-selected 500-top-ranked genes using ReliefF [107] technique. These genes are then used by our proposed iBPSO for optimization process. The parameters used in the proposed iBPSO are listed in Table 3.2. Here the parameter settings was adopted by conducting many trials to get best objective value and inline with many related work which utilizes Binary PSO [23, 108].
Figure 3.3: Total number of genes present in each microarray dataset.

Figure 3.4: Number of genes selected from total genes by three classifiers for BPSO method on six microarray datasets.
3.5: Experimental Setup and Results

Chapter 3: iBPSO gene selection method

Figure 3.5: Number of genes selected from total genes by three classifiers for iBPSO method on six microarray datasets.

Figure 3.6: A comparative bar graph representing number of genes selected by different methods using three classifiers on six microarray datasets.
Figure 3.7: Classification accuracy obtained by three classifiers without gene selection on six microarray datasets.

Figure 3.8: Classification accuracy obtained by three classifiers for BPSO method on six microarray datasets.
3.5: Experimental Setup and Results

Chapter 3: iBPSO gene selection method

Figure 3.9: Classification accuracy obtained by three classifiers for iBPSO method on six microarray datasets.

Figure 3.10: A comparative bar graph of classification accuracy obtained by different methods using three classifiers on six microarray datasets.
Table 3.2: Parameters for improved Binary PSO algorithm

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Swarm Size ((W))</td>
<td>60</td>
</tr>
<tr>
<td>Total Number of Iterations ((T))</td>
<td>100</td>
</tr>
<tr>
<td>(w_{\text{max}})</td>
<td>0.9</td>
</tr>
<tr>
<td>(w_{\text{min}})</td>
<td>0.4</td>
</tr>
<tr>
<td>(c_1)</td>
<td>2</td>
</tr>
<tr>
<td>(c_2)</td>
<td>2</td>
</tr>
<tr>
<td>(v_{\text{min}})</td>
<td>-6</td>
</tr>
<tr>
<td>(v_{\text{max}})</td>
<td>6</td>
</tr>
</tbody>
</table>

Table 3.3: Comparative performance evaluation for six microarray datasets based on Naive-Bayes classification method

<table>
<thead>
<tr>
<th>Datsets</th>
<th>Number of ()original genes</th>
<th>Accuracy (%) without gene selection</th>
<th>BPSO Accuracy (%) #Genes</th>
<th>iBPSO Accuracy (%) #Genes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNS</td>
<td>7129</td>
<td>61.66</td>
<td>53.33</td>
<td>2233</td>
</tr>
<tr>
<td>ALL-AML</td>
<td>7129</td>
<td>74.82</td>
<td>100.00</td>
<td>572</td>
</tr>
<tr>
<td>Breast</td>
<td>24481</td>
<td>52.33</td>
<td>54.63</td>
<td>1930</td>
</tr>
<tr>
<td>Lymphoma</td>
<td>4026</td>
<td>75.42</td>
<td>95.45</td>
<td>948.8</td>
</tr>
<tr>
<td>MLL</td>
<td>12582</td>
<td>70.87</td>
<td>95.8</td>
<td>4721</td>
</tr>
<tr>
<td>SRBCT</td>
<td>2308</td>
<td>64.94</td>
<td>99.00</td>
<td>794</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td>66.67</td>
<td>83.04</td>
<td>1866.46</td>
</tr>
</tbody>
</table>

Table 3.4: Comparative performance evaluation for six microarray datasets based on 1-NN classification method

<table>
<thead>
<tr>
<th>Datsets</th>
<th>Number of ()original genes</th>
<th>Accuracy (%) without gene selection</th>
<th>BPSO Accuracy (%) #Genes</th>
<th>iBPSO Accuracy (%) #Genes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNS</td>
<td>7129</td>
<td>56.66</td>
<td>58.68</td>
<td>2190</td>
</tr>
<tr>
<td>ALL-AML</td>
<td>7129</td>
<td>62.21</td>
<td>95.56</td>
<td>734</td>
</tr>
<tr>
<td>Breast</td>
<td>24481</td>
<td>52.63</td>
<td>55.33</td>
<td>2049</td>
</tr>
<tr>
<td>Lymphoma</td>
<td>4026</td>
<td>80.33</td>
<td>91.52</td>
<td>890</td>
</tr>
<tr>
<td>MLL</td>
<td>12582</td>
<td>59.27</td>
<td>92.11</td>
<td>4812</td>
</tr>
<tr>
<td>SRBCT</td>
<td>2308</td>
<td>50.44</td>
<td>93.82</td>
<td>774</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td>60.26</td>
<td>81.17</td>
<td>1908.16</td>
</tr>
</tbody>
</table>
Table 3.5: Comparative performance evaluation for six microarray datasets based on SVM classification method

<table>
<thead>
<tr>
<th>Datsets</th>
<th>Number of original genes</th>
<th>Accuracy (%) without gene selection</th>
<th>BPSO Accuracy (%)</th>
<th>#Genes</th>
<th>iBPSO Accuracy (%)</th>
<th>#Genes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNS</td>
<td>7129</td>
<td>65.40</td>
<td>68.32</td>
<td>2549</td>
<td>83.33</td>
<td>207</td>
</tr>
<tr>
<td>ALL-AML</td>
<td>7129</td>
<td>65.28</td>
<td>92.05</td>
<td>655</td>
<td>98.61</td>
<td>118</td>
</tr>
<tr>
<td>Breast</td>
<td>24481</td>
<td>52.58</td>
<td>47.03</td>
<td>2289</td>
<td>91.44</td>
<td>213</td>
</tr>
<tr>
<td>Lymphoma</td>
<td>4026</td>
<td>69.70</td>
<td>95.45</td>
<td>770</td>
<td>97.33</td>
<td>70</td>
</tr>
<tr>
<td>MLL</td>
<td>12582</td>
<td>65.28</td>
<td>96.31</td>
<td>4792</td>
<td>98.61</td>
<td>105</td>
</tr>
<tr>
<td>SRBCT</td>
<td>2308</td>
<td>64.94</td>
<td>99.00</td>
<td>650</td>
<td>98.79</td>
<td>120</td>
</tr>
<tr>
<td>Average</td>
<td>63.86</td>
<td>83.03</td>
<td>1950.83</td>
<td>93.02</td>
<td>138.83</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.6: Comparative evaluation of gene selection for six microarray datasets by three classifiers for iBPSO method

<table>
<thead>
<tr>
<th>Datsets</th>
<th>Number of original genes</th>
<th>Gene Selected(%) by iBPSO-NB</th>
<th>Gene Selected(%) by iBPSO-1-NN</th>
<th>Gene Selected(%) by iBPSO-SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNS</td>
<td>7129</td>
<td>0.021</td>
<td>0.026</td>
<td>0.03</td>
</tr>
<tr>
<td>ALL-AML</td>
<td>7129</td>
<td>0.013</td>
<td>0.021</td>
<td>0.016</td>
</tr>
<tr>
<td>Breast</td>
<td>24481</td>
<td>0.007</td>
<td>0.007</td>
<td>0.009</td>
</tr>
<tr>
<td>Lymphoma</td>
<td>4026</td>
<td>0.016</td>
<td>0.008</td>
<td>0.017</td>
</tr>
<tr>
<td>MLL</td>
<td>12582</td>
<td>0.007</td>
<td>0.009</td>
<td>0.008</td>
</tr>
<tr>
<td>SRBCT</td>
<td>2308</td>
<td>0.059</td>
<td>0.055</td>
<td>0.052</td>
</tr>
<tr>
<td>Average</td>
<td>0.0205</td>
<td>0.0210</td>
<td>0.0220</td>
<td></td>
</tr>
</tbody>
</table>

3.5.2 Results and Discussion

The performance of our proposed iBPSO method is examined by comparing the experimental results obtained on six microarray datasets. In Table 3.3, Table 3.4 and Table 3.5, we have compared the classification accuracy obtained without gene selection and, with gene selection by standard BPSO and our proposed iBPSO methods. Table 3.3 shows the experimental results for Naive-Bayes classifier, Table 3.4 for 1-NN classifier and, Table 3.5 for SVM classifier. We have applied 5-fold cross-validation to evaluate the performance of selected gene subsets using three learning approaches. These methods are implemented using MATLAB 2012 environment.

Our experimental results show that the classification accuracy obtained after using gene selection methods is better than without gene selection. The average highest classification accuracy obtained by our proposed iBPSO-NB, iBPSO-1-NN and, iBPSO-SVM are 93.98, 90.58, and 93.02, respectively. The average number of selected genes by the proposed iBPSO method with NB, 1-NN, and SVM are 120.83, 133.66, 138.83 respectively. The percentage of selected genes in each dataset using our iBPSO approach with all three classifiers are summarized in...
Table 3.6. It can be seen that Naive-Bayes classifier gives the lower average of percentage of selected genes than other two classifiers. For ALL-AML (Leukemia) dataset, iBPSO-NB achieved 100% classification accuracy using only 1.4% genes selected whereas BPSO achieved 100% accuracy with 8% genes selected. This demonstrate that our iBPSO achieved better optimization on the number of selected genes without affecting the classification accuracy.

Fig. 3.3, 3.4 and 3.5 provide bar graphs for all six datasets representing their original number of genes, genes selected by BPSO-NB, BPSO-1NN and BPSO-SVM and number of genes selected by our iBPSO-NB, iBPSO-1NN and iBPSO-SVM. A comparative bar graph is provided in Fig. 3.6 to demonstrate the effectiveness of gene selection by our iBPSO over BPSO. These experimental results show that all the six microarray datasets have noisy and irrelevant genes which are not necessary for the classification purpose and also degrade the performance of all the classifiers. Fig. 3.7 gives the accuracy achieved by classifiers without gene selection. In Fig. 3.8 and 3.9 accuracy obtained by the classifiers after applying BPSO and iBPSO is presented graphically on each dataset. Fig. 3.10 illustrates the graphical comparison of classification accuracy achieved by all the approaches. It shows the efficacy of our iBPSO in achieving better classification accuracy over BPSO. For CNS dataset, iBPSO achieved 83.33% accuracy with only 150 genes whereas BPSO degrades the accuracy to 53.33% with 2233 genes. Similarly, for Breast dataset, iBPSO-SVM overlaps the accuracy BPSO-SVM accuracy results because BPSO has lower accuracy than the accuracy achieved without gene selection by SVM.

Fig. 3.11 shows the graphical comparison of average classification accuracy of these three classifiers for iBPSO method on six microarray datasets. The experimental results show that the proposed method effectively improves the classification accuracy and selects a lower dimensional subset of genes. Also, the results of Naive-Bayes classifier are better than the results of other two classifiers.

In our proposed iBPSO, we implemented a probabilistic rule to solve the local stagnation problem of the standard BPSO and also compared the performance to standard BPSO approach. The results show that the proposed method effectively escapes from a local minima stagnation. The reason behind this improvement is that, in standard BPSO, each particle navigates the search space on the basis of its acquired knowledge (pbest) and the globally acquired knowledge
### Conclusion

In this chapter, an improved BPSO approach have been presented for gene selection in cancer classification of microarray gene expression data. The problem of high dimensionality of microarray data has been considered, which affects the predictive accuracy of a classifier due to the presence of noisy and irrelevant data. An improved BPSO is used to implement gene selection and three classifier approaches namely Naive-Bayes, 1-NN, and SVM are used as an evaluators of iBPSO for six microarray cancer classification problems. The method also solves the problem of local stagnation in standard BPSO. Experimental results show that the method effectively selects a small number of genes having high predictive capability and classifies the samples significantly better than the other techniques. However, in case of cancer microarray data, the major goal is to design a classification model capable of learning and adapting to new data in an automatic manner, since the limited number of data samples are available usually and

![Comparison of Classification Accuracies for all methods](image)

Figure 3.11: Comparison of accuracy performance of three classifiers for iBPSO method on six microarray datasets.

$(g_{best})$ by the whole swarm, to find a near-optimal solution. However, if $(g_{best})$ does not change its value continuously, then it affects the exploration capability of the whole swarm.

---

*Thesis Title: “Pattern Recognition using Soft Computing Techniques in Microarray Data Analysis”*
also considering that the database will be extended through time.