Feature Selection Based on Modified Bat Algorithm

Bin YANG†, Yuliang LU†, Kailong ZHU†, Guozheng YANG†, Jingwei LIU††, Nonmembers, and Haibo YIN†, Member

SUMMARY The rapid development of information techniques has lead to more and more high-dimensional datasets, making classification more difficult. However, not all of the features are useful for classification, and some of these features may even cause low classification accuracy. Feature selection is a useful technique, which aims to reduce the dimensionality of datasets, for solving classification problems. In this paper, we propose a modified bat algorithm (BA) for feature selection, called MBAFS, using a SVM. Some mechanisms are designed for avoiding the premature convergence. On the one hand, in order to maintain the diversity of bats, they are guided by the combination of a random bat and the global best bat. On the other hand, to enhance the ability of escaping from local optimization, MBAFS employs one mutation mechanism while the algorithm trapped into local optima. Furthermore, the performance of MBAFS was tested on twelve benchmark datasets, and was compared with other BA based algorithms and some well-known BPSO based algorithms. Experimental results indicated that the proposed algorithm outperforms than other methods. Also, the comparison details showed that MBAFS is competitive in terms of computational time.

key words: feature selection, wrapper model, bat algorithm, premature convergence, SVM

1. Introduction

Machine learning is widely used to learn knowledge from empirical data. Machine learning aims to establish a model that could be applied in data classification [1], weather prediction [2], etc. Nowadays, more and more real world datasets with large numbers of features, which called high-dimensional dataset, are produced. Learning task in high-dimensional dataset is more difficult because of the difficulty of learning process and the learned model’s lower generalization performance. In fact, many of the features in high-dimensional datasets are irrelevant and redundant. Obviously, these features are not useful for the learning model, and may even have a negative effect on the model’s performance.

A common way to deal with such problem is the feature selection (FS) technique [3]. The purpose of FS is to select a subset of relevant features from the original features. This technique reduces the dimensional of datasets by only selecting the relevant features which could be helpful to improve the classification accuracy. Meanwhile, this reduction of features enables to accelerate the learning process and avoids overfitting. Because of the benefit of FS, it has been used widely throughout many fields, including face recognition [4], [5], text classification [6], [7], and medical diagnosis [8], [9].

The existing FS methods can be classified into the filter approach and the wrapper approach on the basis of search strategies of feature selection. The filter approach mainly constructs the subset of features based on the characteristics of datasets without considering a special learning method. Unlike the filter approach, the wrapper approach employs a search strategy to construct the feature subsets. At the same time, a special learning method is employed in the wrapper approach to evaluate the quality of feature subsets. Therefore, the wrapper approach can often achieves better solutions than the filter one [10].

As the wrapper approach’s results mainly rely on the search strategy, various search methods have been designed for the construction of feature subsets, such as sequential forward search [11], sequential backward search [12], and bidirectional selection [13]. Since meta-heuristic methods have the ability to find near-optimal solutions rapidly in the full search space, many studies have been proposed to solve the problems of feature selection by using meta-heuristic methods. For instance, Yong et al. [10] extends a bare bones particle swarm optimization algorithm (BPSO) for feature selection with binary variables, called the binary BPSO. Kabir et al. [14] presented a hybrid ant colony algorithm (ACO) for feature selection, called the ACOFS. The ACOFS emphasizes not only the selection of features but also the attainment of a reduced number of them. Vieira et al. [15] applied some mechanisms to cope with the premature convergence of the BPSO and then presented a Modified BPSO (MBPSO) algorithm for the feature selection. Forsati R et al. [16] proposed a new feature selection algorithm based on a new variant of ACO, namely enriched Ant Colony Optimization (RACO).

Inspired by the fascinating capability of micro-bats in finding the prey and distinguishing the different insects even in darkness, Yang [17] proposed a new meta-heuristic method named bat algorithm (BA). This approach has the advantages of satisfactory convergence and potential parallelization, and it outperform than some well-known meta-heuristic algorithm in continuous constrained optimization problems through their experimental results. Hence, some
researchers attempt to introduce BA into the problem of feature selection. For instance, inspired by the BPSO, a binary Bat Algorithm (BBA) for feature selection was introduced [18], [19]. Based on the unsupervised learning model and binary bat algorithm, Sylvia Selva Rani et al. [20] proposed a novel method to select a subset of relevant features from unlabelled datasets. Rodrigues et al. [21] presented a novel method to select a subset of relevant features from a small dataset. Thus, the BA-based feature selection algorithms mentioned above mainly guide the bats through the global best solution, which may lead to premature convergence.

To overcome such problem, we propose a modified bat algorithm for feature selection, called MBAFS. To enhance the variety of bats, a random bat and the global best bat are applied together to guide the bats’ exploration. On the other hand, some mechanisms are introduced for the purpose of avoiding premature convergence. The performance of proposed method is evaluated by comparing with some other relevant approaches, where the experimental results show that our MBAFS can achieve better results than other algorithms.

The remainder of this paper is organized as follows. Section 2 gives a brief review of BA and its application on feature selection. Then, the proposed MBAFS-based methodology is described in Sect. 3. Section 4 presents the experimental results with discussion. The conclusion is given in Sect. 5.

2. Related Works

2.1 Bat Algorithm

Bats are fascinating animals and their capability of searching preys by echolocation have attracted many researchers’ attention from other fields. Echolocation works as a type of sonar: bats, mainly micro-bats, emit a loud and short pulse of sound, wait it hits into an object and, after a fraction of time, the echo returns back to their ears [22]. Thus, bats can compute how far they are from an object [23]. Furthermore, the echolocation may even help the bats to distinguish one prey from obstacles in complete darkness.

Based on the behavior of bats, a new meta-heuristic algorithm, called Bat Algorithm was proposed by Yang [17] in 2011. Such approach is modeled by imitating the behavior of bats searching for prey. In order to model this algorithm, Yang has idealized some rules as follows [17]:

1. All bats use echolocation to sense distance, and they can also “know” the difference between prey and obstacles in some magic way;
2. A bat $b_i$ flies randomly with the velocity $v_i$ at position $x_i$ with a fixed frequency $f_{\text{min}}$, varying wavelength $\lambda$ and loudness $A_0$ to search for prey. They can automatically adjust the wavelength (or frequency) of their emitted pulses and adjust the rate of pulse emission $r \in [0, 1]$, depending on the proximity of their target;
3. Although the loudness can vary in many ways, we assume that the loudness varies from a large (positive) $A_0$ to a minimum constant value $A_{\text{min}}$.

In bat algorithm, the $ith$ bat $b_i$ update the velocity and its location using Eqs. (1)–(3), as follows:

$$f_i = f_{\text{min}} + (f_{\text{max}} - f_{\text{min}})\beta$$  \hspace{1cm} (1)
$$v_i(t) = v_i(t-1) + \bar{\lambda} \left[ x_i - x_i(t-1) \right] f_i$$ \hspace{1cm} (2)
$$x_i(t) = x_i(t-1) + v_i(t)$$ \hspace{1cm} (3)

where $f_i$ denotes the frequency of $b_i$. $f_{\text{min}}$ and $f_{\text{max}}$ stand for the minimum and maximum frequency respectively. $\beta$ is a random number within the interval $[0, 1]$. $v_i(t)$ and $x_i(t)$ denote the velocity and location of $j$-dimensional for $b_i$ at time step $t$, respectively.

During the searching process, $b_i$ will emit pulse with large loudness and small frequency. Once $b_i$ has found its prey, the loudness and frequency would be updated as follows:

$$A_i^{t+1} = \alpha \times A_i^t$$ \hspace{1cm} (4)
$$r_i^{t+1} = r_i^t[1 - \exp(-\gamma \times t)]$$ \hspace{1cm} (5)

in which $r_i^t$ and $A_i^t$ respectively denote the pulse emission and loudness at time step $t$. $\gamma$ and $\alpha$ are the algorithm parameters.

Just like many other meta-heuristic algorithms, random walks is employed in BA [24], which would enhance the variability of the possible solutions. Primarily, BA chooses one solution among the best bats, and then employs random walks to generate a new solution for each bat:

$$x_{\text{new}} = x_{\text{old}} + \epsilon \bar{A}(t)$$ \hspace{1cm} (6)

where $\bar{A}(t)$ is the average loudness at certain iteration $t$ and $\epsilon \in [-1, 1]$ is a randomly generated number.

Based on the Equations mentioned above, the pseudo-code of standard Bat Algorithm is presented in Fig. 1.

---

**Fig. 1** Pseudo-code of the standard bat algorithm
2.2 BA-Based Feature Selection

As readers can observe, the standard BA just can be used to solve continuous constrained optimization problems, such as function optimization [25], [26]. However, in case of feature selection, some strategies should be introduced to solve the movement of bats in the n-dimensional Boolean lattice.

Inspired by BPSO, Nakamura et al. [18], [19] firstly proposed a binary version of Bat Algorithm for feature selection, namely BBA. BBA uses a sigmoid function to restrict the position of bats to only binary value:

\[ Si(v_i^j) = \frac{1}{1 + \exp(-v_i^j)} \quad (7) \]

Instead of Eq. (3), the position of bats can be updated as follows:

\[ x_i^j = \begin{cases} 1 & \text{if } Si(v_i^j) > \sigma \\ 0 & \text{otherwise} \end{cases} \quad (8) \]

Where \( \sigma \sim U(0, 1) \). Therefore, each bit of bats can be restricted to only binary values, which can indicate the presence or absence of the features.

Taha et al. [21] proposed a naive bayes-guided Bat Algorithm for feature selection, namely NBBA. Instead of updating velocity for each single feature, the velocity of bats is updated by the combination of previous velocity and the number of different bits between the bat and the global best bat. Hence, if the value is negative, it means that the bat has chosen more features than the global best bat. Otherwise, the bat has less features than that of the global best bat. Meanwhile, the NBBA set a maximum velocity \( V_{\text{max}} \) to restrict the step of movement. Another major difference is the way bat update its position. In the case where the velocity of the \( ith \) bat \( b_i \) is lower or equal to the number of different bits between \( b_i \) and the global best bat, \( b_j \) would copy some random different bits from the global best bat. Otherwise, \( b_j \) will import all features from the global best bat to be the same as the global best bat [21]. Besides, the local searching process of the global best solution is controlled by the average of all bats’ loudness in NBBA.

As far as we know, there are only two feature selection methods based on the Bat Algorithm. Since NBBA and BBA was guided only by the global best bat, they may be bad at exploration and exploitation, which may result in the premature convergence.

3. Description of the Proposed MBAFS

As described in Sect. 2.2, BA-based feature selection methods have a high possibility of trapping into local optima. Therefore, the motivation for MBAFS is to design a global search technique with some introduced mechanisms, aiming at improving the results of BA-based algorithm for feature selection. The details of the proposed modified BA for feature selection is given in the following.

3.1 Definition

Assume two Boolean vectors, namely \( a, b \in \{0, 1\}^k \). Then, their hamming distance can be defined as follow:

\[ \text{dist}(a, b) = \sum_{j=1}^{k} |a_j - b_j| \quad (9) \]

The set of different bits between \( a \) and \( b \) is denoted by \( d_f(a, b) \):

\[ d_f(a, b) = \{ j : a_j \neq b_j \} \quad (10) \]

Obviously, \( d_f(a, b) \) contains the positions of different bits between \( a \) and \( b \).

Moreover, we can define the difference of bits between \( a \) and \( b \), namely \( \text{dif}(a, b) \):

\[ \text{dif}(a, b) = \sum_{j=1}^{k} (a_j - b_j) \quad (11) \]

If the value of \( \text{dif}(a, b) \) is higher (lower) than zero, it means that the number of selected features in \( a \) is more (less) than that of \( b \). If the value of \( \text{dif}(a, b) \) is equal to zero, it means that \( a \) selects more number of features than \( b \). Considering following example that \( a = 1100111100, b = 1110011111 \). Then, the value of hamming distance \( \text{dist}(a, b) \) is 6, and \( d_f(a, b) = \{3, 4, 5, 6, 9, 10\} \). \( \text{dif}(a, b) = -2 \) which means that \( a \) selected two fewer features than \( b \).

3.2 Encoding and Fitness Functions

In MBAFS, each bat is encoded as a binary string and the length of string is determined by the number of dataset features. Accordingly, each bit of bat stands for a feature. For example, if a dataset with \( n \) features, each bat would be represented as a \( n \)-bit string.

\[ x_i = (x_i^1, x_i^2, \ldots, x_i^j, \ldots, x_i^n), \quad i = 1, 2, \ldots, m \quad (12) \]

where \( m \) stands for the size of bats, and the \( x_i^j = \{0, 1\} \) denotes whether the \( j \)th feature is chosen in the feature subset.

Each candidate solution is evaluated by a fitness function defined shown as Eq. (13), where \( \text{accuracy} \) denotes the classification accuracy achieved by a classifier and \( S(x) \) is the number of selected features. From Eq. (13), we can see that the fitness of each bat is determined by two different components and each one is weighted differently. The value of \( \alpha \) would be discussed in the experiment design.

\[ \text{fit}(x) = \alpha \times \text{accuracy} + (1 - \alpha) \times \frac{n - S(x)}{n} \quad (13) \]

3.3 Mechanisms to Avoid Premature Convergence

Two mechanisms are introduced to avoid premature convergence in MBAFS. One is called mutation of bats with a operator to displace the location of bats. The other is the
changing for the update of velocity and position. Instead of
guiding by the global best solution, MBAFS employs a ran-
dom bat and the global best bat to guide the movement of
each bat together.

(1) Mutation of bats
In the case if the fitness of the global best bat \( \hat{x}(t) \) stay stable
during several iterations, it means that the bats perhaps drop
into local optima. In order to avoid such problem, the mech-
anism called mutation of bats is introduced. Like Genetic
algorithm, mutation is introduced to jump out of the local optimum
and search for better solution in the search space.

Usually, the value of critical iteration is around 3 itera-
tions, which determined by \( D_{cr} \). The mechanism to mutate
the position of \( i \) th bat \( x_i \) can be formulized as follows:

\[
x_i^j = \begin{cases} 
-\frac{r \cdot \Delta x^j}{\tau}, & r \leq dr \\
x_i^j, & \text{otherwise} 
\end{cases}, \quad (14)
\]

where \( dr \) is the distance rate, usually, the value is set
to \( 1/n \), and \( r \) is a randomly generated number within the
interval \([0, 1]\).

(2) Update of velocity and position
Since BBA and NBBA guide the bats only by the global best
solution, which can not ensure the diversity of bats. Thus,
they may easily get trapped in a local optima. To overcome
such problem, a random bat and \( \hat{x}(t) \) are employed together
to guide the update of bats in MBAFS. The mechanism is
used to improve the variability of bats, which could help to
get better solution.

The position and velocity of bats are updated on the
basis of \( \hat{x}(t) \) and a random bat, called \( x_r(t) \). The velocity
of each bat is composed of two velocity components, namely
\( v_i^x(t) \) and \( v_i^v(t) \):

\[
v_i(t) = \begin{cases} 
v_i^x(t) = \frac{\tau}{T} \times \text{dist}(\hat{x}(t), x_i(t)) \times \tau \\
v_i^v(t) = w \times v_i^v(t-1) + \text{dist}(\hat{x}(t), x_i(t)) \times f_i \end{cases}, \quad (15)
\]

where \( T \) is the maximum iteration, and \( \tau \in [-1, 1] \) is a random
number. The term \( w \) is the inertia weight, which is used to
control the influence of previous velocity. From Eq. (15),
it is clearly that the value of \( v_i^v(t) \) would be very close to zero
with the increasing of iteration. Then the velocity would be
mainly controlled by the component \( v_i^v(t) \). But, It should
be noted that the velocity would be always composed of
the two velocity components.

According to the velocity, the local search scope of
each bat can be divided into two parts, \( D_{x_i}^x(t) \) and \( D_{x_i}^v(t) \),
which determined by \( v_i^x(t) \) and \( v_i^v(t) \) respectively:

\[
\begin{align*}
D_{x_i}^x(t) &= \begin{cases} 
1, & v_i^x(t) < 0 \\
D_{max}, & v_i^x(t) \geq D_{max} \\
[v_i^x(t)] + 1, & \text{otherwise}
\end{cases} \\
D_{x_i}^v(t) &= \begin{cases} 
1, & v_i^v(t) < 0 \\
D_{max}, & v_i^v(t) \geq \frac{T-t}{T} D_{max} \\
[v_i^v(t)] + 1, & \text{otherwise}
\end{cases} \quad (16)
\end{align*}
\]

in which \( D_{max} \) is the maximum exploration space of bats in
each iteration. Usually, the value of \( D_{max} \) is set to \( n/3 \).

The update of the \( i \)th bat’s position is divided into
two steps. In the first step, in the case that the value of
\( \text{dist}(x_i(t), x_r(t)) \) is less than \( D_{x_i}^x(t) \), the \( i \)th bat would randomly
copy \( D_{x_i}^x(t) \) features from \( x_i(t) \), where the features are located in
\( d_f(x_i(t), x_r(t)) \). Otherwise, the \( i \)th bat will import all the
features from \( x_r(t) \). Second, \( D_{x_i}^v(t) \) and \( d_f(x_i(t), \hat{x}(t)) \)
are adopted to update the result achieved above, as well as the
first step.

3.4 Local Search of Bats

In order to realize the local search process, we introduce

| Step | Description |
|------|-------------|
| 1.   | Initialize the bat population \( X_i \) and \( v_i \), \( i = 1, 2, \ldots, n \). |
| 2.   | For each bat |
| 3.   | Choose a randomly generated \( f_r \), loudness \( A_i \), and pulse rates \( r_i \). |
| 4.   | EndFor |
| 5.   | Evaluate the fitness of all the bats. Find the best bat \( \hat{x} \). |
| 6.   | While \( t < T \) |
| 7.   | For each bat \( X_i \) |
| 8.   | Choose a random bat \( X_r \). |
| 9.   | Generate \( D_{x_i}^x(t) \) and \( D_{x_i}^v(t) \) solutions through Eqs. (15-17); |
| 10.  | Generate new solutions by \( D_{x_i}^x(t) \) and \( D_{x_i}^v(t) \). |
| 11.  | EndFor |
| 12.  | If rand \( > r_i \) |
| 13.  | Select a solution \( X_i \) among the best solutions above; |
| 14.  | Generate a local solution \( X_2 \) around \( \hat{x} \) through; |
| 15.  | Choose the better solution \( X_{br} \) between \( X_1 \) and \( X_2 \); |
| 16.  | EndIf |
| 17.  | If rand \( < A_i \) |
| 18.  | \( f(x_{br}) > f(\hat{x}) \) |
| 19.  | Accept the new solutions \( X_{br} \); |
| 20.  | Increase \( r_i \) and reduce \( A_i \); |
| 21.  | MI++; |
| 22.  | EndIf |
| 23.  | MI = 0; |
| 24.  | EndIf |
| 25.  | EndIf |
| 26.  | If MI = 3 |
| 27.  | Generate a new solution for each bat with mutation operator. |
| 28.  | EndIf |
| 29.  | EndWhile |

Fig. 2 Pseudo-code of MBAFS algorithm
the method described in NBBA, where the loudness stands for the change in number of selected features. e.g. if the value of \( \varepsilon \bar{A}(t) \) is three, it means that the bat will randomly choose three bits to change. It is obviously that the value of loudness play an important role in obtaining a good quality solution. Hence, it is necessary to define the maximum and minimum loudness. In MBAFS, we set the maximum loudness as \( n/5 \) and the value of minimum loudness is 1, as well as NBBA.

The complete proposed algorithm is shown in Fig. 2.

4. Experimental Results and Discussion

The empirical results are presented to compare the performance of MBAFS with some well-known feature selection algorithms, including BBA, NBBA, binary particle swarm optimization (BPSO) [28] and improved binary particle swarm optimization (IBPSO) [29]. In order to better evaluate the performance of MBAFS, all the algorithms are run for 20 times on each dataset with different initial solutions.

4.1 Dataset and Experimental Setting

The experiments have been performed on some well-known datasets collected from UCI machine learning repository [30], including WBCD1, Wine, Australian, Zoo, Vehicle, German, WBCD2, Ionosphere, Lung, Sonar, Hillvalley and Musk1. The detail description of these dataset is available in Table 1.

The parameter setting of all the algorithms, including the population of individuals PN, maximum generation MCN, the maximum value of velocity \( V_{\text{max}} \), critical iteration of trapping into local optima I, the value of inertia weight \( w \) and acceleration factors \( c_1, c_2 \), is showed in Table 2. Specifically, the population is set to 25 while the number of features is less than 50. Otherwise, the number of population is set to 60.

Although the fitness function is defined, the value of \( \alpha \) has not been determined. Usually, \( \alpha \) is set to 0.9 [21] or 1.0 [31]. On the one hand, if the value of \( \alpha \) is 1.0, the objective of the algorithm is to improve the classification accuracy without considering the number of selected features. On the other hand, if \( \alpha < 0.9 \), the algorithm may be focus on the reduction of features instead of classification performance. Hence, the value of \( \alpha \) should be mainly chosen within the interval \( [0.9, 1) \). In order to better determine the value of \( \alpha \), the analysis of Eq. (13) is necessary.

From Eq. (13), we can see that the fitness of each individual would be influenced by the number of dataset features. It is said that if the number of features is too small, the number of selected features would be certainly more influential to the fitness than classification accuracy. Considering following four examples shown in Table 3. The Fs refers to the number of selected features, and Ans stands for the classification accuracy of bat. The Fit in Table 3 means the fitness of bats.

Table 3 clearly shows that if the number of features is very small, the value of \( \alpha \) would impact the value of fitness seriously. For example, B1 is better than B2 while the number of dataset features is 14 and the value of \( \alpha \) is set to 0.9. This situation changes if the value of \( \alpha \) is 0.99. But, if the number of dataset features is 20, the value of \( \alpha \) would never result in this phenomenon. Hence, in the case that the value of \( \alpha \) is very small and smaller dimension of feature, algorithms may choose the solution selected less features with lower accuracy rate. In fact, FS aims to reduce the dimension of feature and achieve high percentage of accuracy. Recall that FS prefers a highly percentage of accuracy when datasets have a relative small number of features. Nevertheless, it seems different for the high-dimension datasets, which FS prefers to choose a better subsets with relative high classification accuracy. Therefore, to balance the classification accuracy and the number of selected features efficiently, the value of \( \alpha \) in our experiments can be set as follows:

\[
\alpha = \begin{cases} 
0.99, & n < 20 \\
0.9, & \text{otherwise} 
\end{cases} \tag{18}
\]

Thus, the value of \( \alpha \) is not set to 1.0 when the number of dataset features is less than 20, which means that the purpose of feature selection algorithm is not just the higher classification accuracy. Besides, in the case that the number of dataset features is more than 20, the value of \( \alpha \) would lead algorithm to the results with higher classification accuracy and less features.
4.2 Comparisons between MBAFS, NBBA, BBA, BPSO and IBPSO

Table 4 and Table 5 show the average number of selected features and the average classification accuracy for each algorithm runs on twelve datasets, respectively. The AVL in Table 4 refers to the average length of selected features. The CA in Table 5 is the average percentage of accuracy produced by libSVM. The SD in Tables 4–5 signify the standard deviation. In order to clarify the difference between MBAFS and IBPSO, Fig. 3, which reports the experimental results of MBAFS and IBPSO in 20 runs, is considered. Figure 4 presents the normalized average fitness of all the algorithms, which shows the overall performance of feature selection algorithms. With the results shown in Tables 4–5 and Figs. 3–4, the following observations can be made as below.

(i) It is surprising to see that each dataset could achieve high accuracy rate of classification with using all features. For example, the classification accuracy of Musk1 dataset was 96.91% without the reduction of features. The main reason for this phenomenon may be the high performance of libSVM. As can be seen from Table 4, MBAFS selected smaller number of features for solving different datasets. For instance, 6.8 features, on average, was selected from a set of 17 features in solving the Zoo dataset. On the other hand, the selected number of features in Musk1, a large-dimensional dataset with 166 features, was reduced to 49.32 in MBAFS. In fact, MBAFS was able to selected a small number of features for all the datasets. According to Table 4, MBAFS could achieve better CA with small number of selected features except for the Vehicle and Musk1 datasets. For example, the average CA was increased from 60.00% with all features to 90.67 with 14.1 selected features for the Lung dataset. Similarly, MBAFS was able to achieve an average CA of 95.59% with 19.7 features, whereas the average CA was 89.42 with 60 features for the sonar dataset. It is similar for other remaining datasets in MBAFS except for the Vehicle and Musk1. The average CA with the process of MBAFS was less than the CA with all features in Vehicle and Musk1 datasets, but no significant difference has been figured out. Furthermore, the number of selected features was reduced drastically for these two datasets.

(ii) The number of selected features in MBAFS was superior to BBA over the WBCD1, Wine, Zoo, Vehicle and German datasets. It seems that the BBA would choose less features than MBAFS in the case of high-dimensional datasets. However, the results of CA indicates that MBAFS would achieve better CA than that of BBA, which means that MBAFS can better balance the CA and the number of selected features. Comparing with the results of BPSO, MBAFS was able to get higher CA and less selected features on most of the datasets except the Sonar dataset, which
BPSO increased 0.42% CA with selecting four more features. For example, for the Zoo dataset, MBAFS got 98.91% CA with selected 6.8 features, while for the BPSO methods this value was recorded 98.51% with 7.5 selected features. From the results of NBBA, IBPSO and MBAFS, it can be observed that MBAFS was able to achieve the best CA over the WBCD1, Wine, WBCD2 and Ionosphere datasets, and it acquired the second best CA over the Australian, Zoo, Vehicle and Lung datasets. As shown in Table 4, MBAFS selected the smallest number of features over WBCD1 and German datasets, and the second smallest number of features was obtained over the Wine, Zoo, WBCD2, Ionosphere, Sonar and Musk1 datasets. Furthermore, it is clearly to reveal that all the results of MBAFS caused relatively small SD, as represented in Tables 4–5, which implies the robustness of MBAFS.

(iii) Although the results of MBAFS was superior on most of the datasets, the results of IBPSO and MBAFS was all very similar. To better understand the difference between IBPSO and MBAFS, we emphatically analyzed the experimental results of each algorithm in each run shown in Fig. 3. As shown in Fig. 3, there were a variety number of results that achieved different percentage of classification accuracy with the same number of features. And, MBAFS and IBPSO evolved same classification accuracy with the same number of selected features in the 20 runs, which may be shown in the same point in the figure. Therefore, the results shown in Fig. 3 may be less than 20 distinct points in the chart. It is clear to see that, in most cases, our proposed method was able to include fewer features and achieve higher classification accuracy than IBPSO over the WBCD2, Ionosphere, Lung, Sonar, Hillvalley and Musk1. Hence, Such conclusion suggests that MBAFS could achieve better solutions than IBPSO on the high-dimension datasets, which including more than 30 features. This phenomenon may be due to the mutation mechanism introduced in MBAFS. In the case that the number of dataset features is less than 30, mutation mechanism may lead bats miss the global optimal solutions. In contrast, IBPSO would achieve the global best solution with checking most of the search space. Thus, the results obtained by IBPSO were better than those of MBAFS. However, situation changes when the number of dataset features is more than 30. Because of the mutation mechanism, our proposed method would improve the algorithm search space greatly. Thus, MBAFS would achieve better results than IBPSO.

(iv) From Fig. 4, we can see that the fitness of MBAFS was superior to BBA and BPSO obviously. Comparing the results of MBAFS with NBBA, it can be seen that the solutions MBAFS achieved were better than that of NBBA except the Lung dataset. Accordingly, the overall performance of our proposed algorithm is better than IBPSO. In the Wine, Australian, Zoo and Vehicle dataset, although the results of MBAFS is not superior to IBPSO, they are quite similar to those of IBPSO, which achieved the second best results. Furthermore, it is clearly to observe that, in other datasets with large number of features, the
performance of IBPSO was worse than those of most of other four algorithms. Therefore, we can conclude that the proposed method can achieve better performance than IBPSO, NBBA, BBA and BPSO.

4.3 Comparisons on Computational Time

In order to evaluate the amount of required running time for the proposed feature selection algorithm, the comparison of computational time was reported in Table 6.

We can observe that, for the datasets with small features, the perform of MBAFS took more time than others. The main reason for this phenomenon is the operation of mutation mechanism. During the processing of low dimensional datasets, better solutions may be achieved just through a few iterations, so MBAFS would run the mutation mechanism multiple times. Therefore, MBAFS would consumed more time than other algorithms. However, all algorithms could perform one run within relatively short time, just a few minutes, such as WBCD1, Wine, Zoo, WBCD2, Ionosphere, Lung and Sonar datasets. Moreover, it is interesting to see that, for the datasets with large number of features, MBAFS used less time than IBPSO and BPSO. For example, in the Hillvalley datasets, MBAFS performed one run in 43.32 minutes, whereas for the BPSO and IBPSO methods this value is recorded 54.75 minutes and 63.83 minutes. Recall that in high-dimensional datasets, it is more important to reduce the computational time than the small datasets. Therefore, we can conclude that MBAFS was able to achieve better results within relatively short time over all the datasets.

4.4 Discussions

Experimental results show that our proposed algorithm can be successfully used to reduce the dimension of features and improve the classification accuracy effectively. As we know that the main difficulty to solve the feature selection problems is the local optima which may cause the phenomenon namely premature convergence. To deal with such problem, some mechanisms are employed to maintain the diversity of bats and enhance the ability of jumping out local optima. As discussed in Sect. 3, MBAFS has a new adjustment to guide the update of individuals. Different from other BA-based feature selection techniques, MBAFS employs one random bat to guide the evolution of individuals from generation to generation, which is useful for the exploration of the solution space to search better results. This mechanism is successfully to keep the diversity of bats during generations and further reduce the possibility of converging to local optima. However, there is still a high possibility of encountering local optimization. To enhance the ability of overstep local optima, a mutation operator is introduced. Since the fitness of global best bat stays stable during the premature convergence, the mutation operator would lead to the bats’ convergence to a new search space for better solution. Therefore, MBAFS could usually achieve better solutions than BBA, NBBA, BPSO and IBPSO.

5. Conclusion

In this paper, an improved bat algorithm for feature selection, namely MBAFS, was proposed. To maintain the diversity of bats, an additional bat, which chosen randomly from bats is applied to guide the evolution of bats. Moreover, one important mechanism, namely mutation of bats, was employed to overstep the local optimum effectively.

In order to evaluate the effectiveness of the proposed algorithm, it is compared to some well-known algorithms, including BBA, NBBA, BPSO and IBPSO. Experimental results show that MBAFS can better balance the classification accuracy and the number of selected features over all the twelve datasets. Meanwhile, the low standard deviation of the average number of selected features, as well as the average classification accuracy, means that our proposed algorithm is stable and robust. Furthermore, the comparison on computational time means that MBAFS is able to achieve better results within relatively short time. In the future, we intend to applied MBAFS to solve some complicate problems in some areas.

Acknowledgments

Both authors would like to acknowledge the support of National Natural Science Foundation (NO. 61602491)

References

[1] G. Madelin, F. Poidevin, A. Makrymallis, and R.R. Regatte, “Classification of sodium MRI data of cartilage using machine learning,” Magnetic resonance in medicine, vol.74, no.5, pp.1435–1448, 2015.
[2] S. Lou, D.H.W. Li, J.C. Lam, and W.W.H. Chan, “Prediction of diffuse solar irradiance using machine learning and multivariable regression,” Applied Energy, vol.181, pp.367–374, 2016.
[3] S. Tabakhi and P. Moradi, “Relevance-redundancy feature selection based on ant colony optimization,” Pattern Recognition, vol.48, no.9, pp.2798–2811, 2015.
[4] L. Zini, N. Noceti, G. Fusco, and F. Odone, “Structured multi-class feature selection with an application to face recognition,” Pattern Recognition Letters, vol.55, pp.35–41, 2015.
[5] M. Bennamoun, Y. Guo, and F. Sohel, “Feature selection for 2D and 3D face recognition,” Wiley Encyclopedia of Electrical and Electronics Engineering, 2015.
[6] K. Javed, S. Maruf, and H.A. Babri, “A two-stage Markov blanket
based feature selection algorithm for text classification,” Neurocomputing, vol.157, pp.91–104, 2015.

[7] J.-C. Lamirel, P. Cuxac, A.S. Chivukula, and K. Hjalaloui, “Optimizing text classification through efficient feature selection based on quality metric,” Journal of Intelligent Information Systems, vol.45, no.3, pp.379–396, 2015.

[8] Y. Cong, S. Wang, J. Liu, J. Cao, Y. Yang, and J. Luo, “Deep sparse feature selection for computer aided endoscopy diagnosis,” Pattern Recognition, vol.48, no.3, pp.907–917, 2015.

[9] T.W. Rauber, F. de Assis Boldt, and F.M. Varejão, “Heterogeneous feature models and feature selection applied to bearing fault diagnosis,” IEEE Trans. Ind. Electron., vol.62, no.1, pp.637–646, 2015.

[10] Y. Zhang, D. Gong, Y. Hu, and W. Zhang, “Feature selection algorithm based on bare bones particle swarm optimization,” Neurocomputing, vol.148, pp.150–157, 2015.

[11] A.W. Whitney, “A direct method of nonparametric measurement selection,” IEEE Trans. Comput., vol.100, no.9, pp.1100–1103, 1971.

[12] T. Marill and D. Green, “On the effectiveness of receptors in recognition systems,” IEEE Trans. Inf. Theory, vol.9, no.1, pp.11–17, 1963.

[13] R. Caruana and D. Freitag, “Greedy Attribute Selection,” ICML, pp.28–36, 1994.

[14] M.M. Kabir, M. Shahjahan, and K. Murase, “A new hybrid ant colony optimization algorithm for feature selection,” Expert Systems with Applications, vol.39, no.3, pp.3747–3763, 2012.

[15] S.M. Vieira, L.F. Mendonça, G.J. Fariha, and J.M.C. Sousa, “Modified binary PSO for feature selection using SVM applied to mortality prediction of septic patients,” Applied Soft Computing, vol.13, no.8, pp.3494–3504, 2013.

[16] R. Forsati, A. Moayedikia, R. Jensen, M. Shamsfard, and M.R. Meybodi, “Enriched ant colony optimization and its application in feature selection,” Neurocomputing, vol.142, pp.354–371, 2014.

[17] X.-S. Yang, “A new meta-heuristic bat-inspired algorithm,” Nature inspired cooperative strategies for optimization (NICO SO 2010), pp.65–74, Springer Berlin Heidelberg, 2010.

[18] R.Y.M. Nakamura, L.A.M. Pereira, K.A. Costa, D. Rodrigues, J.P. Papa, and X.-S. Yang, “BBA: a binary bat algorithm for feature selection,” 2012 25th SIBGRAPI Conference on Graphics, Patterns and Images, IEEE, pp.291–297, 2012.

[19] D. Rodrigues, L.A.M. Pereira, R.Y.M. Nakamura, K.A.P. Costa, X.-S. Yang, A.N. Souza, and J.P. Papa, “A wrapper approach for feature selection based on bat algorithm and optimum-path forest,” Expert Systems with Applications, vol.41, no.5, pp.2250–2258, 2014.

[20] A.S.S. Rani and R.R. Rajalaxmi, “Unsupervised feature selection using binary bat algorithm,” 2015 2nd International Conference on Electronics and Communication Systems (ICECS), IEEE, pp.451–456, 2015.

[21] A.M. Taha, A. Mustapha, and S.D. Chen, “Naive bayes-guided bat algorithm for feature selection,” The Scientific World Journal, 2013, 2013.

[22] D.R. Griffin, F.A. Webster, and C.R. Michael, “The echolocation of flying insects by bats,” Animal behavior, vol.8, no.3, pp.141–154, 1960.

[23] W. Metzner, “Echolocation behavior in bats,” Science Progress (1933–), pp.453–465, 1991.

[24] X.S. Yang, “Bat algorithm for multi-objective optimization,” International Journal of Bio-Inspired Computation, vol.3, no.5, pp.267–274, 2011.

[25] S. Yilmaz and E.U. Kucukilles, “A new modification approach on bat algorithm for solving optimization problems,” Applied Soft Computing, vol.28, pp.259–275, 2015.

[26] I. Fister, Jr, D. Fister, and I. Fister, “Differential evolution strategies with random forest regression in the bat algorithm,” Proceedings of the 15th annual conference companion on Genetic and evolutionary computation, ACM, pp.1703–1706, 2013.

[27] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” IEEE Trans. Evol. Comput., vol.6, no.2, pp.182–197, 2002.

[28] S. Lee, S. Soak, S. Oh, W. Pedrycz, and M. Jeon, “Modified binary particle swarm optimization,” Progress in Natural Science, vol.18, no.9, pp.1161–1166, 2008.

[29] L.-Y. Chuang, H.-W. Chang, C.-J. Tu, and C.-H. Yang, “Improved binary PSO for feature selection using gene expression data,” Computational Biology and Chemistry, vol.32, no.1, pp.29–38, 2008.

[30] C. Blake and C.J. Merz, “[UCI] Repository of machine learning databases,” 1998.

[31] N.K. Sreeja and A. Sankar, “Pattern matching based classification using ant colony optimization based feature selection,” Applied Soft Computing, vol.31, pp.91–102, 2015.
Guozheng Yang received the B.E. degree in Communication Engineering from Electronic Engineering Institute, Hefei, P.R. China, in June 2011 and the M.E. degree in Department of Communication & Information from Electronic Engineering Institute in June 2014. He received the Ph.D. degree in Communication & Information Engineering from Electronic Engineering Institute. He is currently a lecturer in the Department of Networks at Electronic Engineering Institute, Hefei, P.R. China. His research interests include complex network theory, data mining and machine learning.

Jingwei Liu received the B.E. degree in Radar Engineering from University of Science and Technology, Nanjing, P.R. China, in June 2012. He is an assist engineer in China Electronic System Engineering Company. Also, he is currently pursuing the M.E. degree with the Communication & Information System, Electronic Engineering Institute. His research interests include Network situation awareness, Game theory and machine learning.

Haiibo Yin received the B.E. degree in Information Engineering from Electronic Engineering Institute, Hefei, P.R. China, in June 2011 and the M.E. degree in Communication & Information System from Electronic Engineering Institute in June 2014. He is currently pursuing the Ph.D. degree with the Department of Communications, Electronic Engineering Institute. His research interests include communication specific emitter identification, intelligent computing, data mining and machine learning.