NATURAL EXTENSIONS: BAT ALGORITHM WITH MEMORY

[1]AHMED MAJID TAHA, [2]SOONG-DER CHEN, [3]AIDA MUSTAPHA
[1]Soft Computing and Data Mining Center, Universiti Tun Hussein Onn Malaysia 86400 Parit Raja, Batu Pahat Johor, Malaysia.
[2] College of Information Technology, Universiti Tenaga Nasional, 43000 Kajang, Selangor, Malaysia
[3] Soft Computing and Data Mining Center, Universiti Tun Hussein Onn Malaysia 86400 Parit Raja, Batu Pahat Johor, Malaysia.
E-mail: [1]ahmed.majid.taha@gmail.com, [2]chensoong@uniten.edu.my, [3]aidam@uthm.edu.my

ABSTRACT

Bat Algorithm (BA) has recently started to attract a lot of attention as a powerful search method in various machine learning tasks including feature selection. Feature selection is essentially a dimensionality reduction problem that aims at two key objectives; to shorten the computational time and to improve classification accuracy. This paper attempts to address the issue of high computational time in feature selection by proposing a natural extension to the BA called the Bat Algorithm with Memory (BAM). This algorithm is inspired from observations that natural bats must rely upon spatial memory in navigating familiar spaces with dimensions larger than a few meters due to their limited biosonar operating range. Using the same approach, bats in existing BA is extended with memory capability to enable them to navigate easily over familiar locations. To evaluate the proposed algorithm, a series of experiments were carried out on twelve datasets with different number of objects and attributes. Next, the experimental results were compared with the original version of BA. The results showed that BAM was able to deliver competitive classification accuracy with increased saving time ranging from 28% up to 95%. The results also demonstrated that the time saving was attributed to three characteristics, which are samples number, feature number, and dataset geography. Consequently, BAM is also more efficient with lower number of features and higher number of samples.

Keywords: *Computational Intelligence, Evolutionary Algorithms, Bioinspired Computing, Feature Selection.*

1. INTRODUCTION

The feature selection problem is concerned with finding the most influential subset of predictors in predictive modeling from a much larger set of potential predictors that can contain hundreds of features. Recently, feature selection has been successfully employed to solve classification problem

in various areas, such as pattern recognition, machine learning and signal processing [1-4]. In real life, not all available features in the datasets are relevant for the classification process. Most of the available features are redundant and add very little, if any, to the prediction power of the classification system. On the other hand, using all the features from the datasets often results in strong over-fitting during classification but with very poor predictions.

Feature selection problem belongs to the realm of combinatorial optimization where the main objective is to find the subset of variables that optimize the value of some goodness of fit function. Because in nature datasets are of high dimensionality, feature selection is part from the group of NP-hard problems [5-7].

This means when the number of potential predictors/features, *k*, is large, the selection process may not be solved within an acceptable amount of computational time. Searching the exact solution is equivalent to checking all $O(k)^2$ states because we have to compare the solutions exhaustively to find the best solutions. Even with medium size *k*, this task is practically impossible. In addition, looking at each predictor's contribution in isolation will not produce accurate results as it ignores the inter-correlations between the predictors. As the result, researches often resort to metaheuristic algorithms since no analytical solution is suitable for the feature selection problem. metaheuristic algorithms offer reasonably good solutions without having us explore the entire solution space. This will lead to a small rick whereby there is no guarantee for the algorithm to find global optimal solutions or even bounded solutions [8, 9].

The organization of this paper keeps on as follows, section 2 presents the background problem and related work. Section 3 begins by introducing the preliminaries on the BA and its application in feature selection. Next, this section also presents the proposed Bat Algorithm with Memory (BAM). Section 4 reports the experimental setup, Section 5 discusses the experimental results and finally, Section 6 concludes the paper.

2. RELATED WORK

Metaheuristic approach to feature selection can be categorized into two types; employed Single Solution-based Metaheuristics (SBM) and Population-based Metaheuristics (PBM). SBM manipulate and transform a single solution using search algorithms such as the Hill Climbing [10] Simulated Annealing [11], and Tabu Search [12], On the other hand, PBM relies on optimization algorithms such as the Genetic Algorithm [13-15], Ant Colony Optimization [16, 17] and Particle Swarm Optimization [18, 19]. metaheuristic optimization algorithms represent an iterative improvement in a population of solutions that works as follows. First, the population is initialized. Then, a new population of solutions is generated. Next, the new population is integrated into the existing population using some selection procedures. The search process will be terminated when a certain criteria has been satisfied.

Both SBM and PBM have a fair balance of advantages and disadvantages. SBM algorithms suffer from two major disadvantages. It often converges towards local optima and it may be very sensitive to the initial solution. Nonetheless SBM algorithms are fast because they are exploitation-oriented, unlike PBM algorithms that are more exploration-oriented. This means PBM algorithms are good at avoiding local optima; hence resulting in higher quality solution but at the expense of time. Efficiency and effectiveness are two conflicting criteria in designing metaheuristic solutions because it is an inverse relationship between time and solution quality, respectively. In most cases, an increase in efficiency will result in a decrease in effectiveness.

More recently, a new metaheuristic algorithm called the Bat Algorithm (BA) was proposed by Yang [20] based on the echolocation behavior of bats. BA has been vigorously applied in various optimization domains, which is often multi-objective and multidisciplinary with complex constraints [21-28] numerical optimization problems [29-33], complicated problems in power system optimization [34-37], image processing [38-41], clustering analysis [42, 43], feature selection [44-46]. BA has shown excellent results and often outperformed other optimization algorithms because it possesses advantages from three different algorithms, which are particle swarm, harmony search, and simulated annealing. Nonetheless, although BA is highly effective, it has low efficiency to match its ability in producing quality solutions.

One aspect worth to investigate in effort to improve efficiency in BA is by observing natural capabilities in real life bats. According to Barchi et al (2013) real-life bats have an excellent spatial memory that allows them to navigate familiar locations easily due to their limited operating range of biosonar. The type of information stored in their spatial memory is the navigation paths of familiar spaces with dimensions larger than a few meters. That means the bats, despite having poor eyesight, rely on mental maps of places they have been to before and use those additional information in their memory to complement their sonar information during navigation [47]. The study reported that bats will still remember their experimental lab and how to fly safely within the room even after three months of experience. In this paper, we propose a natural extension to BA by furnishing the algorithm with memory capability. This will allow the bats to navigate faster in familiar areas or previously visited places.

Memory-based technique in metaheuristics, however, is not a new breakthrough. It was pioneered by Glover in Tabu Search (TS) back in 1989 [48]. The memory allows the algorithms to store information related to the search process, which a specific feature of TS. Tabu search may also be viewed as a dynamic transformation of the neighborhood. This policy may generate cycles; that is, previous visited solutions could be selected again. To avoid cycles, TS discards the neighbors that have been previously visited by memorizing recent search trajectory, which is called the Tabu list. This Tabu list works serves as the short term memory in TS. In subsequent study such as Melián (2006) inspired TS approach for using memory to

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

improves variable neighborhood search and can summarized by changing an algorithm beehive relying on memory [49]. It can observe that TS approach has major disadvantage which may cause escaping from attractive unvisited solutions or areas.

3. BAT ALGORITHM WITH MEMORY

The main objective of the proposed Bat Algorithm with Memory (BAM) is to reduce computational time while maintaining the solutions quality in Bat Algorithms. This is essentially a study to increase performance efficiency of the algorithm with the same equal effectiveness.

3.1 Preliminaries On Bat Algorithm (Ba)

In nature, most micro-bats have advanced capability of echolocation controlled by their extraordinary big auricle. Their very loud but short sonar pulses reflect back from the surrounding objects, producing a flight path based on the echo. Not only they are able to discriminate direction for their own navigation, they are also able to distinguish different types of insects and obstacles while hunting prey, hence intelligently avoiding collision whether during the day or night.

The formalization of BA is as follows. Bats fly randomly with velocity V_i at position x_i with a fixed frequency $f_{\min}$, varying wavelength λ, and loudness A_0 to search for prey. They can automatically adjust the wavelength (or frequency) of their emitted pulses as well as the rate of pulse emission $r \in [0, 1]$, depending on the proximity of their target. Although the loudness varies in many ways, we assume that the loudness varies from a large (positive) A_0 to a minimum constant value A_{min}. The frequency f that lies within the range of $[f_{min}, f_{max}]$ corresponds to a range of wavelengths $[\lambda_{\min}, \lambda_{\max}]$. The frequency may be varied while fixing the wavelength λ, assuming $f \in [0,_{\max}]$ because λ and f are related since $\lambda f = V$ is constant. In simulations, the virtual bats will define the updated rules of their positions x_i and velocities V_i in a D dimensional search space. The new solutions x_i^t and velocities v_i^t at time step t are given by the following equations,

$$f_i = f_{\min} + (f_{\max} - f_{\min}) \quad (1)$$

$$v_i^t = v_i^{t-1} + (x_* - x_i^t)\, f_i \quad (2)$$

$$x_i^t = x_i^{t-1} + v_i^t \quad (3)$$

where $\beta \in [0, 1]$ is a random vector drawn from a uniform distribution. Here, x_* is the current global best location (solution) which is located after comparing all the solutions among all the n bats. For the local search part, once a solution is selected among the current best solutions, a new solution for each bat is generated locally using random walk in the following equation,

$$x_{new} = x_{old} + £ A^t \quad (4)$$

where $\varepsilon \in [-1, 1]$ is a random number and $A_t = (A_i^t)$ is the average loudness of all the bats at this time step. In addition, the loudness A_i and the rate r_i of pulse emission have to be updated accordingly as the iterations proceed. These formulas are:

$$A_i^{t+1} = \sim A_i^t \quad (5)$$

$$r_i^{t+1} = r_i^0\, [1 - \exp(-¥t)] \quad (6)$$

where α and γ are constants.

3.2 Bat Algorithm For Feature Selection

The bat's frequency is represented as a real number while the velocity of each bat is represented as a positive integer number. Velocity suggests the number of bat's attributes that must be changed at certain moment of time. During simulation, the bats communicate with each other via the global best solution and move towards the global best position (solution). The difference between length of global best bat and length of the *i*th bat is illustrated as follows. When the difference is positive, this means the global best bat has more features than those of the *i*th bat. When this happens, the result is summed with previous velocities and the algorithm will accelerate the *i*th bat towards the global best bat. If the difference is negative, this means the *i*th bat has more features than the global best bat. Therefore, when the output is summed with the previous velocity, it will decrease the velocity of *i*th bat and help to attract it closer to global best bat. Each bat's position is formulated as a binary string of length *N*, where *N* is the total number of features. Each feature is represented by bit, where '1' means the corresponding feature is selected and the '0' means it is not selected. The positions are categorized into two groups according to the bit difference between the *i*th bat and the global best bat in order to align exploitation and exploration during searching.

The bat's position is adjusted depending on one of the following conditions; in the case where the velocity of i^{th} bat is lower or equal to the number of different bits, i^{th} bat will copy some features from global best bat, thus moving towards global best bat, while still exploring new search space. In the case where the velocities of i^{th} bat is higher than the velocity of global best bat, then the i^{th} bat will import all features from global best bat to be same as the global best bat with a few different bits to facilitate further exploitation. Loudness A_i is represented as the change in number of features at certain time during local search around the global best bat, as well as local search around the i^{th} bat. The value for sound loudness also plays an important role in obtaining good quality solutions within reasonable amount of time. The choice of the maximum and minimum loudness depends on the domain of application as well as the size of dataset.

Pulse rate r_i has the role to decide whether a local search around the global best bat solution should be skipped or otherwise. Higher pulse rate will reduce the probability of conducting a local search around the global best and vice versa. Therefore, when the bat approaches the best solution, pulse rate value will increase and subsequently reduce the chances to conduct a local search around the global best. Each candidate solution will be evaluated using a NB classifier. Equation 7 defines the fitness function where *P(Y_J |X)* is the classification accuracy and $|C|$ is the total number of features. δ and φ are two parameters corresponding to the weight of classification accuracy and subset length, where $\delta \in [0, 1]$ and $\varphi = 1- \delta$. This equation shows that the importance of classification accuracy and subset size are weighted differently. Generally, classification accuracy is given more weight than the size of subset. In this experiment, the two parameters have been set as follow: $\delta = 0.9$, $\varphi = 0.1$.

$$Sol_A = \text{¢} . P(Y_j \big| X) + \text{¨} . \frac{|C| - |R|}{|C|} \quad (7)$$

3.3 Proposed Extension To Bat Algorithm

Based on previous observation from the behaviors of BA in feature selection, each single bat is able to generate up to three solutions whereby one of the solutions is elective and the other two solutions are compulsory. The first compulsory solution is generated after adjusting the frequency and velocity for each bat. Then the second solution is electively generated around the global best and relies on the rate value. If the pulse rate is high, the probability of generating such solution will be low. Otherwise the solution generation probability will be high. The last compulsory solution is generated around each i^{th} bat position. As the result, generated solutions for BA show higher number of solutions if compared against the traditional algorithms such as PSO and HS. Because more solutions refer to the fitness functions, more calculations are required hence increasing the computational time in BA.

In order to overcome this shortcoming, an extension of memory capability to BA has been proposed for the purpose of speeding up the algorithm without jeopardizing the effectiveness of the algorithm. Nonetheless, memory mechanism in other metaheuristic optimization research such as [12, 50-52] will lead to change in behavior of the algorithm either by guiding the algorithm to avoid local optima or by taking the algorithm to search over more promising area through generation of new solutions based on previous reference set of solutions. This type of memory capability is inappropriate for BA since the bats are already able to identify the promising area individually using their own loudness and pulse rate values. Therefore, manipulation the memory it could speed up the bats' movements without impacting their directions.

Figure 1 illustrates the basic procedure of the proposed BAM, the shaded region represent the main difference from previous proposed BANB. In the proposed BAM, global memory that represents the bat position by the bits '10101011' and its fitness value are included in the swarm. When the bats navigate through the search space, they will check if the certain position has been visited by any bat. The i^{th} bat will send the certain position to the proposed global memory and an inspection process will be conducted to check out whether this position exists or not. If the position has been visited before by any other bats, the global memory will provide i^{th} bat the fitness value for this certain position so the bat is able to navigate easily over this position without taking time to calculate the fitness function.

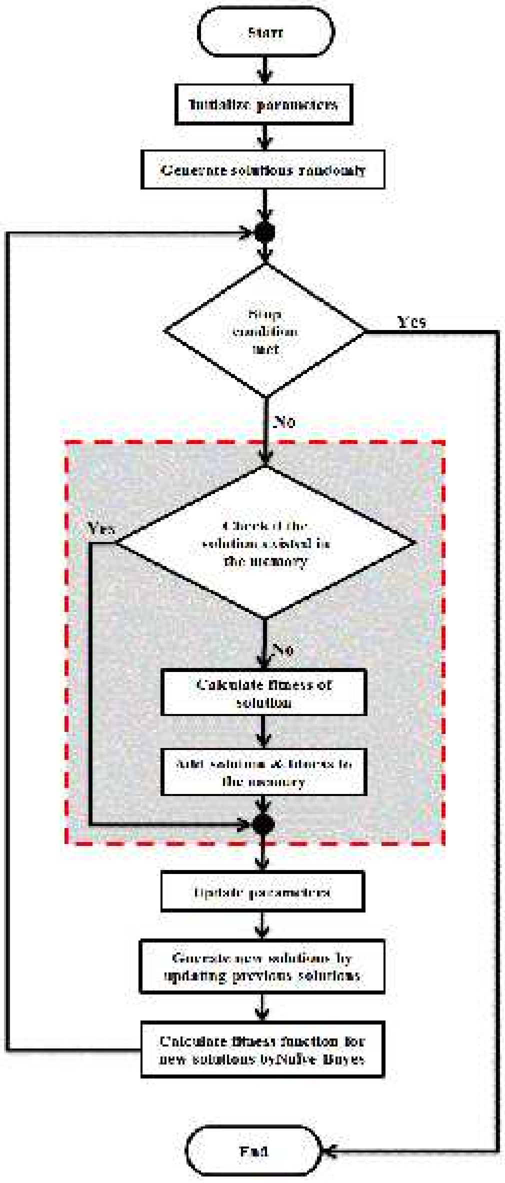
Start
Initialize parameters
Generate solutions randomly
Stop condition met
Yes
No
Check if the solution visited in the memory
Yes
No
Calculate fitness of solution
Add solution & fitness to the memory
Update parameters
Generate new solutions by updating previous solutions
Calculate fitness function for new solutions by Naïve Bayes
End

Figure 1: Proposed Bat Algorithm with Memory

If the position has never been visited before and it is being discovered for the first time, the i^{th} bat will calculate the fitness function for this position and provide the global memory with the solution details. Later, all bats could fly easily over this position by retrieving the information from the global memory. Accordingly, BAM is able to provide the same quality of solutions but at higher efficiency than the BANB.

4. EXPERIMENTAL SETUP

The objective of the experiments is to evaluate the efficiency of the proposed Bat Algorithm with Memory (BAM) versus traditional bat algorithms. To achieve this objective, twelve benchmark datasets from various domains have been selected. Each dataset has different number of features and samples as shown in Table 1.

Both algorithms were run for 30 times, where each time consists of 250 iterations. Table 2 shows the averaged results over 30 runs, whereby MS refers to the number of times BAM uses assembly

Table 1: Characteristics of Datasets

Datasets	No. of features	No. of samples
Lung	56	32
WQ	38	521
Derm2	34	358
Derm	34	366
LED	24	2000
Mushroom	22	8124
Credit	20	1000
Vote	16	300
Heart	13	294
Exactly2	13	1000
Exactly	13	1000
M-of-N	13	1000

memory, FC refers to the number of times the finesses functions were calculated for different solutions, BAM refers to the amount of time consumed by BAM, while BA refers to the amount of time consumed by traditional Bat Algorithm (BA). To ensure fair comparison, the parameters were set to be the same for both algorithms, whereby the population size = 25, decrease/increase sound loudness pulse rates are set to 0.6, and the initial value of pulse rate is equal to 0.2.

Next, in order to investigate the significance of enhancement in term of time consumed, a set of statistical tests were carried out. The results were verified by Kolmogorov—Smirnov and Levene tests, whereby the outcome shows that only some of the data met the assumptions of normality distribution and equality of variance, while the rest did not. Because of this, *T*-test was used for normal data and the Wilcoxon-test for non-normal data. For both tests, the *P*-value is considered to be statistically significant at less than 0.05 and highly significant less than 0.01. Table 3 presents the results from statistical tests for both BAM and BA. Between the brackets is the algorithm that outperformed the other.

5. DISCUSSIONS

Table 2 are show a superior performance by the proposed BAM when BAM outperformed the traditional BA with lower processing time across all datasets except the Lung dataset. Meanwhile, the time saving varies from one dataset to another, the dataset Exactly1 and Exactly2 have the same number of features and samples. Their results were very close to each other's, whereby the time saving was about 95% and they used the memory about fifteen thousand times. Although the M-of-N dataset has the same number of features and samples with Exactly1 and Exactly2, their time saving is lower, which is about 89%.

The Heart dataset has same number of feature as compared to Exactly1 and Exactly2 datasets, but it has different number of samples with saving time at 91%, it consider less compared to Exactly1 and Exactly2 datasets. Vote dataset has higher number of features but almost the same number of samples, this the saving time is 84% and use of memory decreased as compared to the previously mentioned dataset. In credit dataset, the number of features increased but the saving time decreased to 69%. This is acompanied with the decrease in number of memory used as well.

For the mushroom dataset, the difference in time consumption between BAM and BA is the highest among all datasets as illustrated in Figure 2. This is because the mushroom dataset has highest number of samples. Results for saving time were about 67%, which greatly differs from previous results. Since this dataset has higher number of features, thus the saving time decreased. In led dataset, fitness calculation was increased, memory used decreased together with the saving time, down to about 64%. Derm and Derm2 datasets have same number of features and a very small difference in the number of samples, therefore the results for saving time almost same in Derm and Derm2, which are around 39%.

Results for the WQ dataset are shows that the used memory to handle this dataset is the smallest among others datasets due to higher number of features than the traditional BA algorithm. The saving time was about 28%. Although their saving time is considered the smallest, the performance of BAM is still commendable. Finally, the results for Lung dataset deviated compared with the rest data sets, in Figure 2 illustrated the deference in time consumed for the both algorithms performance across all datasets. Table 2 it can be note that traditional BA is faster than the proposed one by 68%. This aberration is justified as the lung dataset has very small number of samples, so searching memory is taking more time for fitness calculation purposes with these samples.

Table 2: Experimental results

Datasets	MU	FC	BAM	BA
M-of-N	14861.83	1610.56	97.16	947.17
Exactly	15792.5	684.6	40.73	859.05
Exactly2	15740.53	692.26	41.27	880.19
Heart	15563.9	1145.4	25.31	291.61
Vote	14327	2004.3	45.79	300.44
Credit	11960.8	4699.36	280.30	927.41
Mushroom	11409.46	4997.2	2494.44	7725.31
LED	10983.93	5358.06	627.85	1789.02
Derm	8564.36	7962.76	229.90	379.88
Derm2	8498.7	8090.3	229.95	370.79
WQ	6855.03	9682.06	401.95	565.70
Lung	6903.23	9514.43	103.32	61.48

Table: 3: Statistical test to evaluate time

Datasets	Significance Test	Test Type
M-of-N	.000 (BAM)	Wilcoxon Test
Exactly	.000 (BAM)	Wilcoxon Test
Exactly2	.000 (BAM)	T-Test
Heart	.000 (BAM)	T-Test
Vote	.000 (BAM)	Wilcoxon Test
Credit	.000 (BAM)	Wilcoxon Test
Mushroom	.000 (BAM)	Wilcoxon Test
LED	.000 (BAM)	T-Test
Derm	.000 (BAM)	Wilcoxon Test
Derm2	.000 (BAM)	T-Test
WQ	.000 (BAM)	T-Test
Lung	.000 (BA)	T-Test

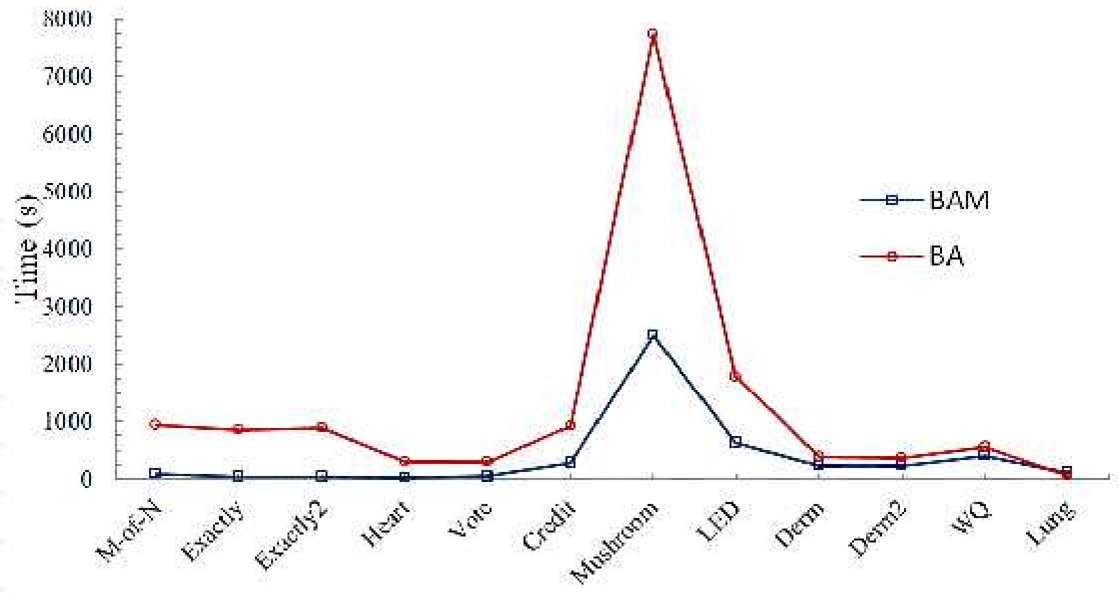

Figure 2: Average time performance across all datasets

In general, the experimental results imply that the proposed BAM is highly affected by three dataset characteristics, which are the type or geography of the dataset, samples number and feature number. This issue may be observed in Exactly1 and M-of-N datasets whereby both have exactly the same number of features and samples but with different saving times. Secondly, with regards to the number of features, it is clear that an increase in number of features corresponds to a decrease in the saving time.

Finally, in terms of the number of samples, the results have shown that the mushroom and Led datasets required different duration of time consumption, whereby the time difference in mushroom dataset was higher (5231s) than the Led dataset (1162s) although Led dataset has higher number of features. BAM is seen to be more efficient with less number of features and higher number of samples. Naive Bayes as a subset evaluator consider very fast compare with others evaluator especially with those belong wrapper methods. Accordingly using evaluator consuming more time to evaluate the subset such as neural network or SVM will resulted in more time saving.

To summarize, the experimental results demonstrated that number of memory used is directly proportional to the increase in time saving while both of them are inversely proportional to number of fitness calculations. The solutions provided by both algorithms have been verified and the quality of the solutions provided was similar. Based on Table 3, the proposed BAM is computationally better than the traditional BA with exception of the lung dataset, where the result was reversed. Finally, the results from statistical test indicated that BAM does not perform well with datasets that consist of very small number of samples.

6. CONCLUSION

In this work, a new bat algorithm extended with memory capability called the Bat Algorithm with Memory (BAM) has been proposed to deal with high computational time issue during feature selection. This problem is resolved by allowing the existing BA to adapt some kind of short term memory in response to mimic the real bats. The proposed BAM was tested on twelve benchmark and well known datasets and then compared against the original BA. The results showed that proposed BAM is capable to provide similar level of accuracy and in fact superior performance as against to the original version. The time saving relied on three dataset characteristics, which are the number of samples, the number features, and dataset geography. Although BAM is found to be more efficient with lower number of features and higher number of samples, the experiments showed that it is impractical to handle datasets with very small number of samples.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

BAM offers many opportunities for further investigation, for example comparative experiments across multiple domains, hybrid approach with other metaheuristic optimization algorithms such as the Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO) in feature selection, effects of different size of datasets where the number of features are a lot higher or evaluate BAM with different classifiers. It is also possible to apply the concept of memory capability in bats to other optimization problems.

REFERENCES:

[1] W. Jialei, Z. Peilin, S. C. H. Hoi, and J. Rong, "Online Feature Selection and Its Applications," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 26, no. 3, pp. 698-710, 2014.

[2] Q. Dai, J.-H. Cheng, D.-W. Sun, and X.-A. Zeng, "Advances in Feature Selection Methods for Hyperspectral Image Processing in Food Industry Applications: A Review," *Critical Reviews in Food Science and Nutrition*, vol. 55, no. 10, pp. 1368-1382, 2015/08/24, 2014.

[3] Y. Yan, H. Shen, G. Liu, Z. Ma, C. Gao, and N. Sebe, "GLocal tells you more: Coupling GLocal structural for feature selection with sparsity for image and video classification," *Computer Vision and Image Understanding*, vol. 124, no. 0, pp. 99-109, 7//, 2014.

[4] S. Oreski, and G. Oreski, "Genetic algorithm-based heuristic for feature selection in credit risk assessment," *Expert Systems with Applications*, vol. 41, no. 4, Part 2, pp. 2052-2064, 3//, 2014.

[5] E. Amaldi, and V. Kann, "On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems," *Theoretical Computer Science*, vol. 209, no. 1, pp. 237-260, 1998.

[6] A. Skowron, and C. Rauszer, "The discernibility matrices and functions in information systems," *Intelligent Decision Support*, pp. 331-362: Springer, 1992.

[7] T. Naghibi, S. Hoffmann, and B. Pfister, "Convex approximation of the NP-hard search problem in feature subset selection." pp. 3273-3277.

[8] E.-G. Talbi, *Metaheuristics: From Design to Implementation*, p.^pp. 624: Wiley, 2009.

[9] F. Min, Q. Hu, and W. Zhu, "Feature selection with test cost constraint," *International Journal of Approximate Reasoning*, vol. 55, no. 1, Part 2, pp. 167-179, 1//, 2014.

[10] R. Caruana, and D. Freitag, "Greedy Attribute Selection." pp. 28-36.

[11] J. W. Debuse, and V. Rayward-Smith, "Feature Subset Selection within a Simulated Annealing Data Mining Algorithm," *Journal of Intelligent Information Systems*, vol. 9, no. 1, pp. 57-81, 1997/07/01, 1997.

[12] A.-R. Hedar, J. Wang, and M. Fukushima, "Tabu search for attribute reduction in rough set theory," *Soft Computing*, vol. 12, no. 9, pp. 909-918, 2007.

[13] J. Yang, and V. Honavar, "Feature subset selection using a genetic algorithm," *Intelligent Systems and Their Applications, IEEE*, vol. 13, no. 2, pp. 44-49, 1998.

[14] A. Jain, and D. Zongker, "Feature selection: Evaluation, application, and small sample performance," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 19, no. 2, pp. 153-158, 1997.

[15] R. Leardi, "Application of a genetic algorithm to feature selection under full validation conditions and to outlier detection," *Journal of Chemometrics*, vol. 8, no. 1, pp. 65-79, 1994.

[16] L. Ke, Z. Feng, and Z. Ren, "An efficient ant colony optimization approach to attribute reduction in rough set theory," *Pattern Recognition Letters*, vol. 29, no. 9, pp. 1351-1357, 2008.

[17] R. Jensen, and Q. Shen, "Fuzzy-rough data reduction with ant colony optimization," *Fuzzy Sets and Systems*, vol. 149, no. 1, pp. 5-20, 1/1/, 2005.

[18] X. Wang, J. Yang, X. Teng, W. Xia, and R. Jensen, "Feature selection based on rough sets and particle swarm optimization," *Pattern Recognition Letters*, vol. 28, no. 4, pp. 459-471, 2007.

[19] A. Unler, and A. Murat, "A discrete particle swarm optimization method for feature selection in binary classification problems," *European Journal of Operational Research*, vol. 206, no. 3, pp. 528-539, 11/1/, 2010.

[20] X.-S. Yang, "A New Metaheuristic Bat-Inspired Algorithm Nature Inspired Cooperative Strategies for Optimization

(NICSO 2010)," Studies in Computational Intelligence J. González, D. Pelta, C. Cruz, G. Terrazas and N. Krasnogor, eds., pp. 65-74: Springer Berlin / Heidelberg, 2010.

[21] X.-S. Yang, and A. H. Gandomi, "Bat Algorithm: A Novel Approach for Global Engineering Optimization," *Engineering Computations*, vol. 29, no. 5, 2012.

[22] T. C. Bora, L. d. S. Coelho, and L. Lebensztajn, "Bat-Inspired Optimization Approach for the Brushless DC Wheel Motor Problem," *IEEE Transactions on Magnetics*, vol. 48, no. 2, pp. 947-950, 2012.

[23] O. Hasançebi, and S. Carbas, "Bat inspired algorithm for discrete size optimization of steel frames," *Advances in Engineering Software*, vol. 67, no. 0, pp. 173-185, 1//, 2014.

[24] R. Mallick, R. Ganguli, and M. S. Bhat, "Robust design of multiple trailing edge flaps for helicopter vibration reduction: A multi-objective bat algorithm approach," *Engineering Optimization*, pp. 1-22, 2014.

[25] T.-T. Nguyen, C.-S. Shieh, M.-F. Horng, T.-G. Ngo, and T.-K. Dao, "Unequal Clustering Formation Based on Bat Algorithm forWireless Sensor Networks," *Knowledge and Systems Engineering*, Advances in Intelligent Systems and Computing V.-H. Nguyen, A.-C. Le and V.-N. Huynh, eds., pp. 667-678: Springer International Publishing, 2015.

[26] P. Dash, L. C. Saikia, and N. Sinha, "Automatic generation control of multi area thermal system using Bat algorithm optimized PD–PID cascade controller," *International Journal of Electrical Power & Energy Systems*, vol. 68, no. 0, pp. 364-372, 6//, 2015.

[27] J. Sadeghi, S. M. Mousavi, S. T. A. Niaki, and S. Sadeghi, "Optimizing a bi-objective inventory model of a three-echelon supply chain using a tuned hybrid bat algorithm," *Transportation Research Part E: Logistics and Transportation Review*, vol. 70, no. 0, pp. 274-292, 10//, 2014.

[28] S. Yılmaz, and E. U. Küçüksille, "A new modification approach on bat algorithm for solving optimization problems," *Applied Soft Computing*, vol. 28, no. 0, pp. 259-275, 3//, 2015.

[29] P. W. Tsai, J. S. Pan, B. Y. Liao, M. J. Tsai, and V. Istanda, "Bat algorithm inspired algorithm for solving numerical optimization problems," *Applied Mechanics and Materials*, vol. 148, pp. 134-137, 2012.

[30] X.-s. He, W.-J. Ding, and X.-S. Yang, "Bat algorithm based on simulated annealing and Gaussian perturbations," *Neural Computing and Applications*, pp. 1-10, 2013/11/23, 2013.

[31] K. Premkumar, and B. V. Manikandan, "Speed control of Brushless DC motor using bat algorithm optimized Adaptive Neuro-Fuzzy Inference System," *Applied Soft Computing*, vol. 32, no. 0, pp. 403-419, 7//, 2015.

[32] X.-s. He, W.-J. Ding, and X.-S. Yang, "Bat algorithm based on simulated annealing and Gaussian perturbations," *Neural Computing and Applications*, vol. 25, no. 2, pp. 459-468, 2014/08/01, 2014.

[33] T.-S. Pan, T.-K. Dao, T.-T. Nguyen, and S.-C. Chu, "Hybrid Particle Swarm Optimization with Bat Algorithm," *Genetic and Evolutionary Computing*, Advances in Intelligent Systems and Computing H. Sun, C.-Y. Yang, C.-W. Lin, J.-S. Pan, V. Snasel and A. Abraham, eds., pp. 37-47: Springer International Publishing, 2015.

[34] B. Ramesh, V. Mohan, and V. Reddy, "Application of bat algorithm for combined economic load and emission dispatch," *Int. J. Electric. Electron. Eng. Telecommun*, vol. 2, pp. 1-9, 2013.

[35] S. ARUNA, and T. D. RAJU, "COMBINED ECONOMIC LOAD AND EMISSION DISPATCH EVALUTION USING BAT ALGORITHM," *nternational Recognition Multidisciplinary Research Journals*, vol. 3, no. 5, 2013.

[36] M. H. Khooban, and T. Niknam, "A new intelligent online fuzzy tuning approach for multi-area load frequency control: Self Adaptive Modified Bat Algorithm," *International Journal of Electrical Power & Energy Systems*, vol. 71, no. 0, pp. 254-261, 10//, 2015.

[37] M. R. Sathya, and M. Mohamed Thameem Ansari, "Load frequency control using Bat inspired algorithm based dual mode gain scheduling of PI controllers for interconnected power system," *International Journal of Electrical Power & Energy Systems*, vol. 64, no. 0, pp. 365-374, 1//, 2015.

Journal of Theoretical and Applied Information Technology
10th September 2015. Vol.79. No.1
© 2005 - 2015 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[38] S. Akhtar, A. Ahmad, and E. Abdel-Rahman, "A Metaheuristic Bat-Inspired Algorithm for Full Body Human Pose Estimation." pp. 369-375.

[39] J. W. Zhang, and G. G. Wang, "Image Matching Using a Bat Algorithm with Mutation," *Applied Mechanics and Materials*, vol. 203, pp. 88-93, 2012.

[40] A. Alihodzic, and M. Tuba, "Improved Bat Algorithm Applied to Multilevel Image Thresholding," *The Scientific World Journal*, vol. 2014, pp. 16, 2014.

[41] Z.-W. Ye, M.-W. Wang, W. Liu, and S.-B. Chen, "Fuzzy entropy based optimal thresholding using bat algorithm," *Applied Soft Computing*, vol. 31, no. 0, pp. 381-395, 6//, 2015.

[42] K. Khan, A. Nikov, and A. Sahai, "A Fuzzy Bat Clustering Method for Ergonomic Screening of Office Workplaces Third International Conference on Software, Services and Semantic Technologies S3T 2011," Advances in Intelligent and Soft Computing D. Dicheva, Z. Markov and E. Stefanova, eds., pp. 59-66: Springer Berlin / Heidelberg, 2011.

[43] M. Sood, and S. Bansal, "K-Medoids Clustering Technique using Bat Algorithm," *International Journal of Applied Information Systems*, vol. 5, pp. 20-22, 2013.

[44] R. Y. M. Nakamura, L. A. M. Pereira, D. Rodrigues, K. A. P. Costa, J. P. Papa, and X.-S. Yang, "9 - Binary Bat Algorithm for Feature Selection," *Swarm Intelligence and Bio-inspired Computation*, X.-S. Yang, Z. Cui, R. Xiao, A. H. Gandomi and M. Karamanoglu, eds., pp. 225-237, Oxford: Elsevier, 2013.

[45] A. M. Taha, and A. Y. C. Tang, "Bat algorithm for rough set attribute reduction," *Journal of Theoretical and Applied Information Technology*, vol. 51, no. 1, 2013.

[46] A. M. Taha, A. Mustapha, and S.-D. Chen, "Naive Bayes-Guided Bat Algorithm for Feature Selection," *The Scientific World Journal*, vol. 2013, no. Recent Advances on Bioinspired Computation, pp. 9, 2013.

[47] J. R. Barchi, J. M. Knowles, and J. A. Simmons, "Spatial memory and stereotypy of flight paths by big brown bats in cluttered surroundings," *The Journal of experimental biology*, vol. 216, no. 6, pp. 1053-1063, 2013.

[48] F. Glover, "Tabu search—part I," *ORSA Journal on computing*, vol. 1, no. 3, pp. 190-206, 1989.

[49] B. Melián, "Using memory to improve the VNS metaheuristic for the design of SDH/WDM networks," *Hybrid Metaheuristics*, pp. 82-93: Springer, 2006.

[50] H. Zhang, and G. Sun, "Feature selection using tabu search method," *Pattern Recognition*, vol. 35, no. 3, pp. 701-711, 3//, 2002.

[51] M. A. Tahir, A. Bouridane, and F. Kurugollu, "Simultaneous feature selection and feature weighting using Hybrid Tabu Search/K-nearest neighbor classifier," *Pattern Recognition Letters*, vol. 28, no. 4, pp. 438-446, 3/1/, 2007.

[52] Y. Wang, L. Li, J. Ni, and S. Huang, "Feature selection using tabu search with long-term memories and probabilistic neural networks," *Pattern Recognition Letters*, vol. 30, no. 7, pp. 661-670, 5/1/, 2009.