# Efficient Data Mining with Evolutionary Algorithms for Cloud Computing Application

Hamid Malmir[1], Fardad Farokhi[2], Reza Sabbaghi-Nadooshan[3]

[1,2,3] Electrical Engineering Department, Central Tehran Branch, Islamic Azad University, Tehran, Iran.
Email: Ham.Malmir.eng@iauctb.ac.ir, F_Farokhi@iauctb.ac.ir, R_Sabbaghi@iauctb.ac.ir

**Abstract**

With the rapid development of the internet, the amount of information and data which are produced, are extremely massive. Hence, client will be confused with huge amount of data, and it is difficult to understand which ones are useful. Data mining can overcome this problem. While data mining is using on cloud computing, it is reducing time of processing, energy usage and costs. As the speed of data mining is very important, this paper proposes four faster classification algorithms in comparison with each other. In this paper, A Multi-Layer perceptron (MLP) Network is trained with Imperialist Competitive Algorithm (ICA), Particle Swarm Optimization (PSO), Differential Evolution (DE), and Invasive Weed Optimization (IWO) separately. The classifications are done on Wisconsin Breast Cancer (WBC) data base. At the end, to illustrate the speed and accuracy of these classifiers, they are compared with each other and two other types of Genetic algorithm classifiers (GA).

*Keywords:* Data Mining, Classification, Cloud Computing, Imperialist Competitive Algorithm, Particle Swarm Optimization, Differential Evolution, Invasive Weed Optimization.



## 1. Introduction

Data mining is extracting useful information from a large number of data that is flaw, noisy and random [1]. Data mining has been successfully used for various applications such as engineering, commerce, military, and so forth.

The main goal of cloud computing are providing virtual resources (such as hardware, software, platforms, etc.) to clients according to their needs [2]. In this case, clients do not need to be professional and know how these resources are configured. It can reduce costs and improve efficiency [3].

When cloud based applications are developed and the amount of data are growing rapidly, data mining is crucial to improve quality of cloud services. For example, YouTube is recommending the new videos, according to analyzing client historical favorites. Salesforce.com needs data mining to provide CRM (Customer Relationship Management) service. University of Chicago and University of Florida initiated Science Clouds project that they called it Nimbus. Nimbus toolkit allows scientific community to lease remote resources by deploying VM (Virtual Machine) [4].

The pioneer IT companies are established their own cloud services. For example, Amazon established Amazon Cloud Drive, Google established Google Drive and Microsoft established OneDrive. Consequently, they can provide numerous services to their client [5].

Jim Gray, Turning Award winner, thought that the amount of data generated every 18 month is equal to the amount of the whole data generated until now. This massive amount of data can confuse clients to understand which information is useful. Cloud computing is a good solution for this problem [6].

Wilding [7] uses Back Propagation (BP) algorithm to train Artificial Neural Network (ANN). It seems that their ANN did not work well. It also uses BP algorithm which can easily trap in a local minimum. As BP is a gradient descend method, it has slow convergence. Tang [8] proposes a combination of

Back Propagation (BP) algorithm and Particle Swarm optimization (PSO) to optimize the parameters of BP neural networks, convergence speed and precision of BP neural networks, but it needs better generalization and much accuracy. Giannella [9] proposes an algorithm that permits to make an efficient decision tree on a heterogeneous distributed database. It uses a random projection-based dot product estimation and message sharing strategy. The benefit of this technique is reducing the communication, but it needs more local computation than centralized algorithm. Wang [10] combines Global Effect (GE), k-nearest neighbors (KNN) and Restricted Boltzmann Machine (RBM) to testify their performance utilizing cloud computing. The experiments show that the KNN was the best algorithm, and GE was after KNN, but RBM was not as well as they thought in RMSE (Root Mean Square Error). Ding [11] proposes a Classification Rules Mining Model with Genetic Algorithm in Cloud computing (CGCRMM). It demonstrated that the decision tree is very slow for massive data mining. Hence, it is not suitable for cloud computing. The benefit of Genetic Algorithm (GA) like other evolutionary algorithms is capability of escaping from local optimum and finding global optimum.

Rest of the paper is structured as follows: Section 2 explains method of classification, ICA, PSO, DE, and IWO algorithms. Section 3 illustrates the results. Section 4 concludes the whole research.

## 2. Method

### *2.1. Classification*

The classification is a very important task in data mining. It extracts a functions or rule which can map each instance in the input vector to one of the given groups at output [12].

ICA, PSO, DE, and IWO are successfully used to optimize many functions and solve application problems [13-28]. This paper proposes to use ICA, PSO, DE, and IWO in order to improve speed and accuracy of classification. They are explained as follow:

### *2.2. Imperialist Competitive Algorithm (ICA)*

Imperialist Competitive Algorithm (ICA) was introduced by Atashpaz and Lucas in 2007 [13]. It starts with some initial countries (like population in GA) that they have random cost. According to their cost (low cost is better), they are classified in two groups: imperialists and colonies. The better countries are considers as imperialists and the other as their colonies. Afterwards, there are three steps: Assimilation, Revolution and Competition. In Assimilation, the imperialists try to absorb their colonies by improving their positions. Revolution changes the position of country suddenly. During movement of colonies toward their imperialist, a colony may reach to a position that has better cost than imperialist, in this case, the position of colony and imperialist will be exchanged. Competition is attempts of imperialists to maintain their colonies and achieve the other countries colonies. Each empire that cannot increase its power or prevent decreasing its power will be eliminated. Finally weaker empires lose their colonies and just the most powerful empire will remain. All the colonies belong to this empire, and they are as powerful as the imperialist. In addition, they have the same position as the imperialist, but they are under the control of the imperialist [13].

The pseudo code for Imperialist Competitive Algorithm is displayed as:

1. Initialize problem and algorithm parameters.
   1.1. Assign variables borders.
   1.2. Assign numbers of country, Imperialists, Iterations.
   1.3. Assign revolution rate, assimilation coefficient, angle of assimilation, zeta, damp ratio, uniting threshold.
2. Initialize random countries and evaluate costs.
3. Determine imperialists and colonies (According their costs).
4. Allocate colonies to imperialists with the probability of $P_n$.
5. Found empires.
6. Assimilation policy: move the colonies toward their imperialist.
7. Revolution: change the position of countries in the Socio-Political Characteristics with the amount of revolution rate.
8. Evaluate costs again.
   8.1. Exchange position of colony and imperialist if it reaches better position than the imperialist.
   8.2. Calculate the total cost for empires.
9. Imperialist Competition: each imperialist tries to achieve the other Imperialists colonies.
   9.1. Choose the weakest colony of the weakest empire, and imperialistic competition will start.
   9.2. The most powerful empire has more chance to gain the colony, but the others have chance too.
   9.3. If one imperialist remains without any colonies, this empire will eliminate.
10. Stop if there is just one empire, and iterations are finished, else go to 6

### *2.3. Particle Swarm Optimization (PSO)*

Particle Swarm Optimization (PSO) introduced by Kennedy, Eberhart and Shi in 1995 is based on a

metaphor of social interaction [14]. This algorithm is successfully used to optimize many continuous functions [14-18]. A number of particles are initialized (like population in GA) with random positions $\overrightarrow{x_i}$ and velocities $\overrightarrow{v_i}$. Function $f$ is defined to evaluate the costs. At first, best position of the best particle will be stored as $\overrightarrow{p_g}$ (Global best), also the position of each particle will be stored as $\overrightarrow{p_i}$ (Local best). The new coordinate of each particle is obtained from the summation of current position of particle, current velocity vector, distance of Local best from current position of particle and distance of Global best from current position of particle. In each iteration, it will be checked whether the new position of each particle is better than its local best or not, then it will be replaced as the local best if it is so, then it will be compared with Global best too. These steps will be repeated until a stop condition satisfied [19].

The new coordinate of a particle is defined by:

$$v_{id}(t+1) = w.v_{id}(t) + r_1.c_1(p_{id} - x_{id}) + r_2.c_2(p_g - x_{id}) \quad (4)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \quad (5)$$

$v_{id}(t+1)$: Velocity vector of next step (Index $d$ shows the dimension of problem).
$v_{id}(t)$: Current velocity vector of $i^{th}$ particle.
$p_{id}$: Best position of $i^{th}$ particle (Local best).
$x_{id}$: Current position of $i^{th}$ particle.
$p_g$: Best position of all particles until now (Global best).
$w$: Inertia weight.
$r_1$ , $r_2$: Uniformly distributed random numbers.

$$r_1, r_2 \sim U(0\,,1) \quad (6)$$

$c_1$: Personal learning ratio.
$c_2$: Population learning ratio.

In this paper, the Constriction Coefficient is used to optimize the performance of PSO [19]. The parameters are defined by:

$$\varphi_1\,,\varphi_2 > 0 \quad , \quad \varphi \triangleq \varphi_1 + \varphi_2 \quad (7)$$

$$\chi = \frac{2k}{\varphi - 2 + \sqrt{\varphi^2 - 4\varphi}} \quad (8)$$

The best values for these parameters are shown as follows:

$$\begin{cases} w = \chi \\ c_1 = \chi.\varphi_1 \\ c_2 = \chi.\varphi_2 \end{cases} \quad (9)$$

### 2.4. Differential Evolution (DE)

The Differential Evolution (DE) introduced by Storn and Price in 1997 to minimize continuous benchmark functions [20]. Like the other evolutionary algorithms, it is a population-based method. It has been successfully used to diverse applications [21-23], and it works better than Genetic Algorithm (GA). Some benefits of DE are: very easy to use, short codes, few parameters to adjust and global minimum search.

DE starts with initialize some random population, and a function to evaluate the population cost. The array is defined as:

$$population = [\,p_1, p_2, p_3, \dots, p_{N_{par}}] \quad (10)$$

$N_{par}$ is the number of problem's dimension.

*2.4.1. Mutation:* the distance between two random individuals "$b$" ,"$c$" will be calculated, and it will be added to random individual's position "$a$" to generate temporary answer named $y$. This procedure is called mutation. It is defined by:

$$y = a + \beta(b - c) \quad (11)$$

$$\beta = [\,\beta_1, \beta_2, \beta_3, \dots, \beta_{N_{par}}] \quad (12)$$

The points $a, b, c$ are not equal. $\beta$ is an uniformly distributed random matrix. Fig. 1 shows the mutation and generation of temporary answer named $y$.

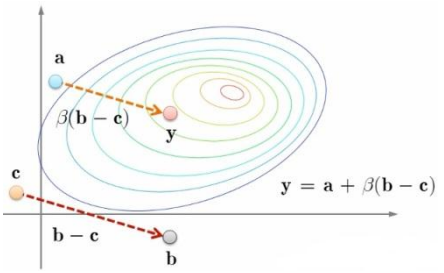


Fig.1. Procedure of mutation and generation of $y$ temporary answer (http://www.matlabsite.ir/).

*2.4.2. Crossover:* the crossover is introduced to increase the diversity of the new answers. The binomial crossover is used in this paper which is defined by:

$$z_j = \begin{cases} y_j & , \quad r_j \le P_{CR} \quad or \quad j = j_0 \\ x_j & , \quad otherwise \end{cases} \quad (13)$$

Index $y_j$ is the temporary answer of the next population. $r_j$ is the uniformly distributed random numbers between [0, 1]. $P_{CR}$ is probability of crossover ratio. $j$ is a number between [1, $N_{par}$]. $j_0$ is a random number between [1, $N_{par}$]. The $x_j$ is the answer of the current population and $z_j$ is the new answer of next population.

*2.4.3. Selection:* In order to determine the members of the next population, the new answer $z_j$ will compare with the current answer $x_j$. Each one which has lower cost will be assigned as a member of the next population.

### *2.5. Invasive Weed Optimization (IWO)*

Invasive Weed Optimization (IWO) introduced by Mehrabian and Lucas in 2006 [24]. This algorithm is successfully used to find global minimum of many benchmark functions, and solve engineering problems [24-28]. The IWO is a numerical stochastic search algorithm imitating colonizing behavior of weeds for function optimization. It includes following steps:

*1) Initialize a Population:* a population with random positions is dispersing over the *d* dimensional problem space.

*2) Reproduction:* each plant produces number of seeds according to its fitness. In this way, the plant with highest fitness (lowest cost) has maximum rate of breeding and vice versa. The number of seeds producing by plants determines with a linear function.

*3) Spatial dispersal:* the generated seeds will scatter around their parents with normal distributions, and mean of these distributions is equal to zero. This will enable algorithm to cover whole search space, and it increases opportunity of finding the best position. The distribution has a standard deviation (SD), $\sigma$, decreasing in a nonlinear manner from an initial value, $\sigma_{initial}$, to a final value, $\sigma_{final}$, in each iteration (generation). The decrement in standard deviation allows plants to reproduce nearer seeds in comparison with prior iterations. The deviation for each generation defines as:

$$\sigma_{iter} = \frac{(iter_{max} - iter)^n}{(iter_{max})^n}\left(\sigma_{initial} - \sigma_{final}\right) + \sigma_{final} \quad (14)$$

$iter_{max}$ is the maximum number of iterations, and $\sigma_{iter}$ is the standard deviation at the current iteration, and $n$ is the nonlinear modulation index.

*4) Competitive exclusion:* as iterations pass, the number of population in colony will pass its maximum acceptable number, $p_{max}$, so this algorithm needs a strategy to maintain the number of population at the maximum allowable number which is called Competitive exclusion. In this case, population is ranking according to fitness, and weeds whose fitness are lower (have higher cost) than others will be eliminated from population to reach the maximum acceptable number.

### *2.6. Data processing*

It is consisting of the following steps:

*1) Data preprocessing:* It includes three steps. First, delete the instances with missing attributes. Second, normalize the attributes into the range of [-1, 1]. Third, randomly divide the instances between train and test.

*2) Training Multi-Layer Perceptron (MLP) Network with ICA, PSO, DE or IWO:* In this step, MLP Network is trained with ICA, PSO, DE or IWO to determine the correct weights.

*3) Testing trained network and evaluating results:* Finally, the test instances are applied on trained network to show how good the MLP Network is trained, and results are plotted.

The flow graph of data processing is shown in Fig. 2.

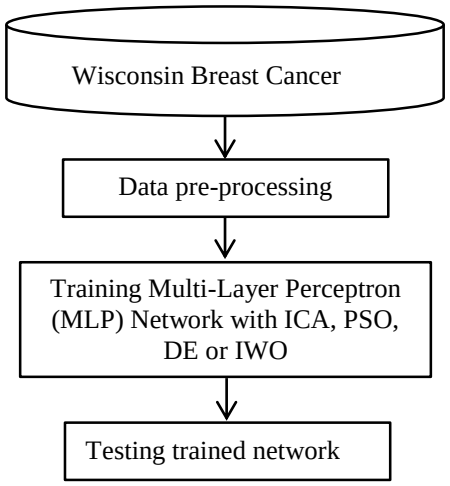


Fig.2. The flow graph of Data processing procedure.

## 3. Results

The neural network's structure has nine input, five hidden neurons and an output. In order to show the performance of the ICA, PSO, DE, and IWO, they are applied to Wisconsin Breast Cancer dataset collected from the UCI [29]. The class attributes are the Breast cancer diagnosis in which 2 means that patient's Breast cancer is benign and 4 indicates malignant Breast cancer. Wisconsin Breast Cancer dataset has 699 instances. Two third of the data are allocated to train, and the rest is used for test. The simulations are

done with MATLAB R2013a on a Core i7 laptop with 8GB ram. The attributes in details are shown in Table 1. The best parameters setting for each algorithm has acquired from numerous runs. They are shown in Tables 2, 3, 4, and 5. The evaluation methods are defined by:

$$Sensitivity = \frac{TP}{TP + FN} \quad (15)$$

$$Specificity = \frac{TN}{TN + FP} \quad (16)$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (17)$$

Table.1
Wisconsin Breast Cancer Dataset Attributes

| No. | Attribute | Domain |
|---|---|---|
| **1** | Sample code number | id number |
| **2** | Clump Thickness | 1 – 10 |
| **3** | Uniformity of Cell Size | 1 – 10 |
| **4** | Uniformity of Cell Shape | 1 – 10 |
| **5** | Marginal Adhesion | 1 – 10 |
| **6** | Single Epithelial Cell Size | 1 – 10 |
| **7** | Bare Nuclei | 1 – 10 |
| **8** | Bland Chromatin | 1 – 10 |
| **9** | Normal Nucleoli | 1 – 10 |
| **10** | Mitoses | 1 – 10 |
| **11** | Class | 2 for benign, 4 for malignant |

Table.2
Parameters Setting of Imperialist Competitive Algorithm (ICA)

| | Parameter | Domain |
|---|---|---|
| **1** | number of countries | 500 |
| **2** | number of imperialists | 30 |
| **3** | number of iterations | 40 |
| **4** | revolution rate | 0.4 |
| **5** | assimilation coefficient (β) | 2 |
| **6** | assimilation angle coefficient (γ) | 0.5 |
| **7** | constant (ζ) | 0.005 |
| **8** | damp ratio | 0.99 |
| **9** | Uniting threshold | 0.02 |

Table.3
Parameters Setting of Particle Swarm Optimization (PSO)

| | Parameter | Domain |
|---|---|---|
| **1** | number of particles | 500 |
| **2** | number of iterations | 40 |
| **3** | constant (k) | 0.8 |
| **4** | Random positive number ($\varphi_1$) | 2.05 |
| **5** | Random positive number ($\varphi_2$) | 2.05 |

Table.4
Parameters Setting of Differential Evolution (DE)

| | Parameter | Domain |
|---|---|---|
| **1** | number of population | 500 |
| **2** | number of iterations | 40 |
| **3** | Beta (β) | 0 - 0.3 |
| **4** | Probability of crossover | 0.4 |

Table.5
Parameters Setting of Invasive Weed Optimization (IWO)

| | Parameter | Domain |
|---|---|---|
| **1** | number of population | 100 |
| **2** | number of iterations | 40 |
| **3** | Maximum number of plant population ($p_{max}$) | 200 |
| **4** | Maximum number of seeds ($S_{max}$) | 10 |
| **5** | Minimum number of seeds ($S_{min}$) | 1 |
| **6** | Nonlinear modulation index ($n$) | 3 |
| **7** | Initial value of standard deviation ($\sigma_{intial}$) | 0.2 |
| **8** | Final value of standard deviation ($\sigma_{final}$) | 0.001 |

Each algorithm is run 10 times with the best parameters setting. Table 6 shows sensitivity, specificity, and accuracy of each method in iteration 20 and 40. It can be observed from Table 7, all proposed algorithms have reached to accuracy of more than 96.43% in generation 20, but Classification Rules Mining Model with Genetic Algorithm in Cloud computing (CGCRMM) and traditional genetic classification (TGC) only have reached to 52% and 51% respectively. CGCRMM and TGC need at least 80 generation to reach the accuracy of 90%. It means that the proposed algorithms are very faster than CGCRMM and TGC; hence, they are suitable for cloud computing where time of data mining is so important. It also shows that at the end, generation 40, Invasive Weed Optimization (IWO) algorithm achieves the Accuracy of 97.99% which is the highest Accuracy in comparison to other algorithms.

Table.6
Experimental Results on Classification for ICA, PSO, DE, and IWO

| | ICA | | PSO | | DE | | IWO | |
|---|---|---|---|---|---|---|---|---|
| **Parameter** | ***20*** | ***40*** | ***20*** | ***40*** | ***20*** | ***40*** | ***20*** | ***40*** |
| **Class2 Sensitivity** | 0.9671 | 0.9679 | 0.9671 | 0.9671 | 0.9749 | 0.9689 | 0.9836 | 0.9771 |
| **Class2 Specificity** | 0.9918 | 0.9945 | 0.9895 | 0.9915 | 0.9838 | 1 | 0.9324 | 0.9850 |
| **Class2 Accuracy** | **0.9760** | **0.9775** | **0.9750** | **0.9763** | **0.9785** | **0.9794** | **0.9643** | **0.9799** |
| **Class4 Sensitivity** | 0.9918 | 0.9945 | 0.9895 | 0.9915 | 0.9838 | 1 | 0.9324 | 0.9850 |
| **Class4 Specificity** | 0.9671 | 0.9679 | 0.9671 | 0.9671 | 0.9749 | 0.9689 | 0.9836 | 0.9771 |
| **Class4 Accuracy** | **0.9760** | **0.9775** | **0.9750** | **0.9763** | **0.9785** | **0.9794** | **0.9643** | **0.9799** |

Table. 7
Experimental Results on Classification Accuracy

| **Algorithm** | **Number of Iterations** | |
|---|---|---|
| | ***20*** | ***40*** |
| IWO-MLP | 96.43% | **97.99%** |
| DE-MLP | 97.85% | 97.94% |
| ICA-MLP | 97.60% | 97.75% |
| PSO-MLP | 97.50% | 97.63% |
| CGCRMM Model | 52% | 72% |
| TGC Model | 51% | 70% |

Table 8 shows that Imperialist Competitive Algorithm (ICA) is the fastest algorithm, and it has the lowest time consumption for classification.

Table. 8
Comparison of Time Consumption for Classification

| **Algorithm** | **Time (sec.)** |
|---|---|
| ICA-MLP | **472.14** |
| IWO-MLP | 516.71 |
| PSO-MLP | 535.41 |
| DE-MLP | 543.08 |

Figs. 3, 4, 5, and 6 illustrate the Correlation of real (supervisor) and network output for test data in ICA, PSO, DE, and IWO algorithms respectively in iteration 40. The best condition is the complete correlation between real and network output. The blue line shows the real output, and the dark circles show network output. Since the dark circles having appropriate correlation with blue line, the network is perfectly trained. Furthermore, it is hard to distinguish between figures, and they look the same because the accuracies of methods are really close to each other, and their differences are less than one percent.

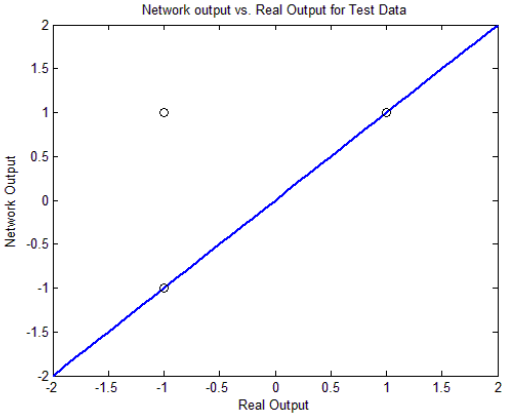


Fig.3. Correlation of real and network output for test data in ICA

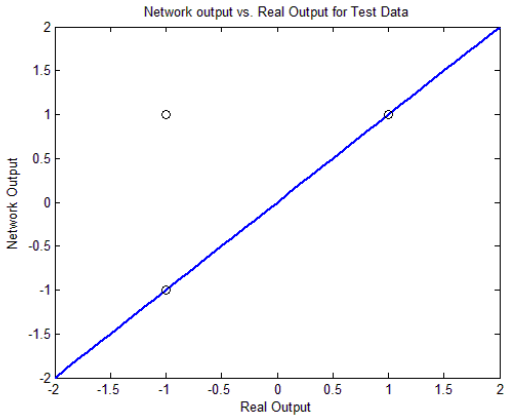


Fig.4. Correlation of real and network output for test data in PSO

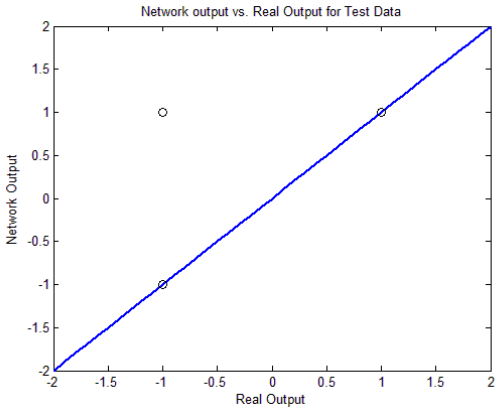


Fig.5. Correlation of real and network output for test data in DE

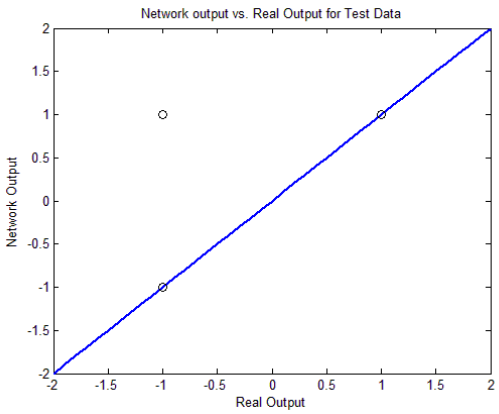


Fig.6. Correlation of real and network output for test data in IWO

## 4. Conclusion

Since the amount of data is increasing continuously on internet, the utilization of data mining is inevitable. Moreover, the speed of data mining is very important. In this paper, the classification is considered, and four evolutionary algorithms are used

to train a neural network. The results demonstrate that IWO is the most accurate algorithm, and ICA is the fastest algorithm in comparison to others; in addition, all proposed algorithms are much faster and accurate than CGCRMM and TGC. All in all ICA not only has rapid operation but also has suitable accuracy; hence it can be a good suggestions for data mining in cloud computing in which it is vital to classify large amount of data in a shorter elapsed time. We are going to work on a big data set on a real cloud infrastructure in order to investigate the mentioned characteristics and to identify how well the evolutionary algorithms can deal with the large data.

## References

[1] C. Jin, SP. DAI and JL. GUO, "Tutorial of artificial intelligence", Beijing: Tsinghua University Press, 2007 (In Chinese).

[2] M. E. GUO and C. Z. XU, "Cloud computing programming model and adaptive resource management", CCCF, Vol. 7, No. 7, pp. 26–33, July 2011 (In Chinese).

[3] P. Liu, "Cloud computing", Beijing: Electronic Industry Press, 2011 (In Chinese).

[4] K. Keahey, R. Figueiredo, J. Fortes, T. Freeman and M. Tsugawa, "Science Clouds: Early Experiences in Cloud Computing for Scientific Applications", in Proc. of High Performance Computing and Communications, 2008.

[5] H. Malmir, F. Farokhi and R. Sabbaghi-Nadooshan, "Optimization of Data Mining with Evolutionary Algorithms for Cloud Computing Application", International Conference on Computer and Knowledge Engineering (ICCKE), pp. 354–358, 2013.

[6] K. Chen and WM. Zheng, "Cloud computing: System instances and current research", Journal of Software, Vol. 20, No. 5, pp. 1337–1348, 2009 (In Chinese).

[7] P. Wilding, M.A. Morgan, A.E. Grygotis, M.A. Shoffner and E.F. Rosato, "Application of backpropagation neural networks to diagnosis of breast and ovarian cancer", Cancer Letter, Vol. 77, No. 2–3, pp. 145–153, March 1994.

[8] P. Tang and Z. Xi, "The Research on BP Neural Network Model Based on Guaranteed Convergence Particle Swarm Optimization", Second Intl. Symp. on Intelligent Information Technology Application, IITA '08, Vol. 2, pp. 13–16, Dec. 2008.

[9] C. Giannella, K. Liu, T. Olsen and H. Kargupta, "Communication efficient construction of decision trees over heterogeneously distributed data", in Proc. of the Fourth IEEE Int. Conf. on Data Mining, pp. 67–74, 2004.

[10] J. Wang, J. Wan, Z. Liu and P. Wang, "Data mining of mass storage based on cloud computing", in Proc. of Ninth Int. Conf. on Grid and Cooperative Computing, 2010.

[11] J. Ding and S.Yang," Classification Rules Mining Model with Genetic Algorithm in Cloud Computing", Int. Journal of Computer Applications, Vol.48, No.18, pp.24–32, June 2012.

[12] T. Hu, H. Chen, L. Huang and X. Zhu, "A Survey of Mass Data Mining Based on Cloud computing", Int. Conf. on Anti-Counterfeiting, Security and Identification (ASID), Taipei, pp. 1–4, Aug. 2012.

[13] E. Atashpaz-Gargari and C. Lucas, "Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition", in IEEE Congress on Evolutionary Computation (CEC), pp. 4661–4667, 2007.

[14] J. Kennedy and R. C. Eberhart, "Particle swarm optimization" , in Proc. IEEE Int. Conf. Neural Networks, Perth, Australia, pp.1942–1948, Nov. 1995.

[15] P. Angeline, "Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences", in Evolutionary Programming VII, V. W. Porto, N. Saravanan, D. Waagen, and A. E.Eiben, Eds. Berlin, Germany: Springer-Verlag, pp. 601–610,1998.

[16] J. Kennedy, "The particle swarm: Social adaptation of knowledge", in Proc. Int. Conf. Evolutionary Computation, Indianapolis, IN, pp. 303–308, Apr. 1997.

[17] J. Kennedy, "Methods of agreement: Inference among the eleMentals", in Proc. 1998 IEEE Int. Symp. Intelligent Control, pp.883–887, Sept. 1998.

[18] Y. Shi and R. C. Eberhart, "Parameter selection in particle swarm adaptation", in Evolutionary Programming VII, V. W. Porto, N. Saravanan, D. Waagen, and A. E. Eiben, Eds. Berlin, Germany: Springer-Verlag, pp. 591–600, 1997.

[19] M. Clerc and J. Kennedy, "The Particle Swarm—Explosion, Stability, and Convergence in a Multidimensional Complex Space", IEEE Transactions on Evolutionary Computation, Vol. 6, No. 1, pp. 58–73, Februrary 2002.

[20] R. Storn and K. Price, "Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces", Journal of Global Optimization, Vol. 11, No. 4, pp. 341–359, 1997.

[21] J. Ionen, J. K. Kamarainen and J. Lampinen, "Differential evolution training algorithm for feed-forward neural networks", Neural Process Letters, Vol. 17, No. 1, pp. 93–105, 2003.

[22] R. Storn, "Designing nonstandard filters with differential evolution", IEEE Signal Processing, Vol. 22, No. 1, pp. 103–106, 2005.

[23] R. Joshi and A. C. Sanderson, "Minimal representation multisensory fusion using differential evolution", IEEE Trans. on Systems, Man, and Cybernetics, Part A: Systems and Humans, Vol. 29, No. 1, pp. 63–76,1999.

[24] A. R. Mehrabian and C. Lucas, "A novel numerical optimization algorithm inspired from weed colonization", Ecol. Inform., Vol. 1, No. 4, pp. 355–366, Dec. 2006.

[25] S. Karimkashi and A. A. Kisk, "Invasive Weed Optimization and its Features in Electromagnetics", IEEE Transactions on Antennas and Propagation, Vol. 58, No. 4, pp. 1269–1278, Apr. 2010.

[26] M.I.K.M. Safari, N.Y. Dahlan, N.S. Razli and T.K.A. Rahman, "Electricity Prices Forecasting Using ANN Hybrid with Invasive Weed Optimization (IWO)", IEEE 3rd International Conference on System Engineering and Technology (ICSET), pp. 275–280, Aug. 2013.

[27] A. Ghosh, A. Chowdhury, R. Giri, S. Das and A. Abraham,"A hybrid evolutionary direct search technique for solving Optimal Control problems", 10th international Conference on Hybrid Intelligent Systems (HIS), pp. 125–130, Aug. 2010.

[28] R. Giri, A. Chowdhury, A. Ghosh, S. Das, A. Abraham, V. Snasel, "A Modified Invasive Weed Optimization Algorithm for training of feed-forward Neural Networks", IEEE International Conference on System Man and Cybernetics (SMC), pp. 3166–3173, Oct. 2010.

[29] H. William, W. Wolberg, N. Street and O. L. Mangesarian, 1992, UCI Machine Learning Repository, http://archive.ics.uci.edu/ml/datasets.html