DOI: https://doi.org/10.31341/jios.44.2.9 UDC 004.02:616.858
Open Access Original Scientific Paper

Quantum-Inspired Evolutionary Algorithms for Neural Network Weight Distribution: A Classification Model for Parkinson's Disease

Srishti Sahni *srishti.sahni16@gmail.com*
Maharaja Agrasen Institute of Technology
New Delhi, India

Vaibhav Aggarwal *aggarwalvaibhav73@gmail.com*
Maharaja Agrasen Institute of Technology
New Delhi, India

Ashish Khanna *ashishkhanna@mait.ac.in*
Faculty of Computer Science
Maharaja Agrasen Institute of Technology

Deepak Gupta *deepakgupta@mait.ac.in*
Faculty of Computer Science
Maharaja Agrasen Institute of Technology

Siddhartha Bhattacharyya *dr.siddhartha.bhattacharyya@gmail.com*
Faculty of Electrical Engineering and Computer Science
VSB Technical

Abstract

Parkinson's Disease is a degenerative neurological disorder with unknown origins, making it impossible to be cured or even diagnosed. The following article presents a Three-Layered Perceptron Neural Network model that is trained using a variety of evolutionary as well as quantum-inspired evolutionary algorithms for the classification of Parkinson's Disease. Optimization algorithms such as Particle Swarm Optimization, Artificial Bee Colony Algorithm and Bat Algorithm are studied along with their quantum-inspired counter-parts in order to identify the best suited algorithm for Neural Network Weight Distribution. The results show that the quantum-inspired evolutionary algorithms perform better under the given circumstances, with qABC offering the highest accuracy of about 92.3%. The presented model can be used not only for disease diagnosis but is also likely to find its applications in various other fields as well.

Keywords: Parkinson's Disease, Particle Swarm Optimization, Artificial Bee Colony Algorithm, Bat Algorithm, Quantum Optimization, Neural Network Weight Distribution.

1. Introduction

Quantum Computers were first designed to provide scientists with a machine to compute results of various physical experiments [5] as they closely simulated the workings of the world we live in. A Quantum Computer replaces a traditional bit with a quantum-bit (qbit), which does not store either 0 or 1 but an overlapping state of the two. This property of the qbit increases the computational capabilities of a quantum computer exponentially. Where n bits could store n units of information, n qbit can store upto 2^n units of information [6]. Quantum Computers are the future of technology and a number of tech-giants are constantly working to make sure this future arrives here in no time but until then, quantum-inspired computing is used to achieve the benefits of a quantum computing device using a regular computer. Quantum-inspired Computing algorithms may use more time for computation as compared to regular algorithms but have proven to be powerful in every fields that they have been applied to [2] [7] [8].

Evolutionary Algorithms are a set of generic population-based metaheuristic optimization algorithms that draw their inspiration from biological processes like evolution, natural selection, genetic mutation, movements of bird flocks and fish schools etc [4]. Evolutionary Algorithms can find their application in a variety of fields and have been used constantly for feature selection in classification models. The proposed model offers a different approach in which Evolutionary Algorithms are used for neural network weight distribution along with their quantum-inspired counter parts in order to draw comparisons between their performances and to identify the best solution to the problem at hand. The said model is used for the classification (diagnosis) of Parkinson's Disease in its early stages and works with 180 data-points.

Degenerative Diseases are a result of continuous degeneration of the cells of the body that affect the tissues, organs and at times the entire organ systems of the patient, which increasingly deteriorates with time. Parkinson's Disease is a degenerative neurological disorder that targets the neurons of the person making it difficult for them to share or transfer even the most basic of information [1]. The early symptoms of the disease include shaking of hands, visible speech impairments and difficulty in walking [2], with speech impairments being the first one to appear. Acoustic tools for speech analysis measure voice functions objectively. Disordered sustained vowels exhibit wide-ranging phenomena, from nearly periodic to highly complex, aperiodic vibrations, and increased "breathiness". The speech of a patient suffering from voice disorders exhibits complex nonlinear aperiodicity, and turbulent, aeroacoustic, non-Gaussian randomness and hence, can only be detected using recurrence and fractal scaling. These techniques are used for accurate measurement of non-linear and non-Gaussian behaviors, the two main biophysical symptoms of disorder and collectively produces a "hoarseness" diagram to depict the disorderliness in the voices of the diseased.

The proposed model is trained on the voice disorder data-set collected by Little M.A., McSharry P.E., Roberts S.J., Costello D.A.E. and Moroz I.M. (2013) [3] which has undergone non-linear Recurrence and Fractal Scaling, and consists of 22 features. A subset of these features is fed to the neural network which then returns the

probability of the presence of the disease. This probability is then used to classify if the person is suffering from the said disease or not.

The proposed model can be used for classification in general and is equipped for solving any given problem with great accuracy.

The highlights of the paper are:

- Use of a multi-layered Perceptron Neural Network for the classification of Parkinson's Disease.
- Use of various evolutionary algorithms such as Particle Swarm Optimization, Artificial Bee Colony Optimization, and Bat Algorithm for Neural Network Weight Distribution.
- Use of Quantum Optimization algorithm for derivation of the quantum counterparts of the said algorithms.
- The model is trained using the above mentioned six algorithms and the results compared in order to identify the best possible solution.
- Quantum-inspired Artificial Bee Colony Algorithm is the most efficient and accurate among the given algorithms with an accuracy of about 92.3%.

  The following section 2 introduces the background of the paper and discusses the various techniques that were combined to formulate the presented solution; Section 3 introduces the quantum counterparts of the algorithms discussed in the preceeding section; Section 4 discusses the observed results, which is then followed by the conclusions and future perspectives for the derived model.

2. Background

2.1. Artificial Neural Network

An artificial Neural Network is the building block of deep-learning models. It is a simple logical architecture that is designed to mimic the workings of a brain i.e. it collects information by observing the surroundings, processes that information over time, draws conclusions about its observations, and stores them for future use [9]. The computational unit of an artificial neural network is an artificial neuron which is modeled after a biological neuron.

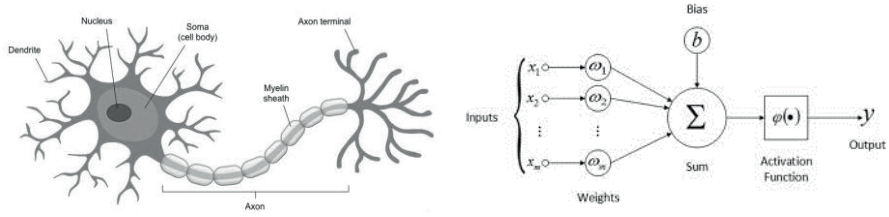

Figure 1. Biological Neuron vs Artificial Neuron

Just like a biological neuron, an artificial neuron receives inputs from various sources, and uses them to modify them using their computational constants (weights). These modified values are then combined and passed through an activation function

which returns its output. Hence, neuron can be defined using 3 parameters: number of inputs, weight matrix and its activation function [10]. The number of inputs and the activation function remain unchanged throughout the lifetime of the neuron but its weight matrix is re-distributed during its testing phase, based on the conclusions drawn by it.

A neural network is defined using its architecture and the neuron which is used as its base unit. The proposed model uses a multi-layered perceptron neural network model with three layers, where the output for the preceding layer is fed as the inputs to the next layer of neurons.

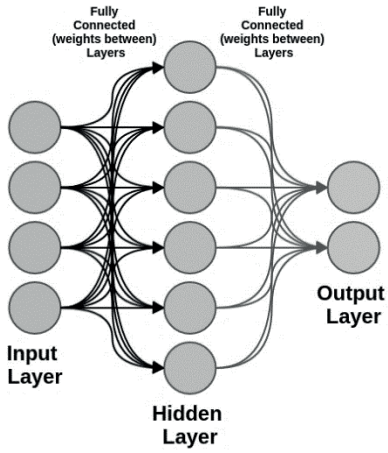

Figure 2. A Multi-layer Perceptron model with three layers

The input to the first layer is the selected subset of features, which are then fed into the second layer which consists of 5 neurons and returns one output. The final output gives the probability of the presence of the disease. The activation function of the proposed network is a sigmoid function that return a value between $(0,0.5)$ if the input lies between $(-\infty,0)$ and returns a value between $(0.5,1)$ if the input lies between $(0,\infty)$, which can be defined as

$$g(x)=1/1+e^{(-x)} \quad (1)$$

2.2. Logistic Regression and Classification

Logistic Regression is a predictive analysis that returns a binary independent variable as an output to the given set of inputs [13]. The proposed model uses logistic regression to predict the presence of Parkinson's disease, such that he output 0 signifies the absence and output 1 signifies the presence of Parkinson's Disease. Logistic regression is implemented with the use of a neural network using sigmoid function as its activation function.

If the output of the neural network is less than 0.5, the value of the binary variable is predicted to be 0. Similarly, if the output value is greater than or equal to 0.5, the value of the binary variable is predicted to be 1. The error in the predictions can be estimated by the following cost function [14]

$$J(\theta) = -1/n \left(\sum_{i=1}^{n} y^{(i)} \times \log(h^{(i)}) + (1 - y^{(i)}) \times \log(1 - h^{(i)}) \right) \quad (2)$$

Here, n is the number of data points, $y^{(i)}$ is the expected value of the binary variable and $h^{(i)}$ is the output of the neural network. The aim of the weiht distributing algorithm is to minimize the cost function, $J(\theta)$, in order to insure greater accuracy of the proposed model.

2.3. Particle Swarm optimization (PSO)

Particle Swarm Optimization is a nature-inspired optimization algorithm that mimics the behavior of a swarm (or flock) of birds and a fish school [15]. The movement of the birds in a bird flock is governed by two principles, its own personal experiences and the collective experiences of the flock it is a member of. The same principles guide the iterations and variations in positions of the birds in the given flock. Unlike any other Evolutionary Algorithm, PSO is not based on selection and hence the entire population survives from the beginning to the end of the training period [16].

The particle is assumed to be present in an n-dimensional vector space where n is the degree of the solution. The position of each particle represents a solution and undergoes variations throughout the course of its iteration in search of the optimum solution [15]. The following equations govern this variation

$$V_i = \omega V_i + c_0 \times rand() \times p_{best}^{(i)} + c_1 \times rand() \times g_{best} \quad (3)$$

$$P_i = P_i + V_i \quad (4)$$

Here, V_i is the velocity of the i^{th} particle, ω is the weight of inertia and ranges from [0, 1], c_0 and c_1 are constants, and rand() generates a random number between (0, 1). P_i is the position of the particle (or the solution in the vector space). $p^{(i)}_{best}$ is a variable that stores the personal best position of the particle and g_{best} stores the global best solution for the entire flock.

The entire algorithm is run itr number of times and the solution with the minimum value of the cost function is stored as $p^{(i)}_{best}$ or g_{best} for the system. The global best solution at the end of itr iterations is selected as the final solution for the given problem.

Particle Swarm Optimization Algorithm

1. *Initialize the bird population P_i and V_i*
2. *Set the values for the constants ω, c_0, c_1*
3. *Initialize the values $p^{(i)}_{best}$ and g_{best}*

4. ***while t < itr***
   a. *For each particle i, calculate the value of the cost function.*
   b. ***If (cost(P_i) < $p^{(i)}_{best}$)***
      $p^{(i)}_{best} = P_i$
      end if
   c. *Find the particle with the minimum cost.*
   d. ***If (cost(P_i) $_{(min)}$ < g_{best})***
      $g_{best} = P_{i\,(min)}$
      end if
   e. *Modify the positions of the particles using equation (3) and (4)*
   end while
5. *Use the final results*

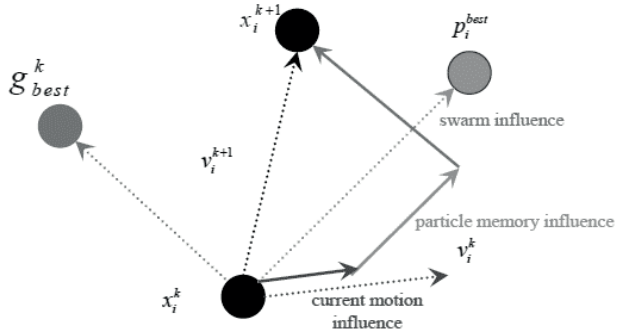

Figure 3. Particle Swarm Optimization

2.4. Artificial Bee Colony (ABC)

Artificial Bee Colony algorithm is based on the foraging behavior of the honey bee swarm. Everyday, the honeybee leaves their hive to go foraging i.e. they go in search of food and collect nectar from the food source based on the quality, amount of nectar available and the distance from the hive [17]. The bee stores the nectar in its stomach and travels back home. It transfers the nectar to the hive and tells the other bees about its conquest in the form of a dance. This dance is a direct representation of the amount of nectar consumed which in turn is a function of its quality and distance of the from the hive [17]. Other bees evaluate the dances of the dancing bees and then determines the best food source.

The bee population is divided into three groups namely; employed bees, onlooker bees, and scout bees.

2.4.1. Employed Bees

An employed bee is assigned to a particular food source and is tasked with the collection from that particular food source and its nearest neighbors. Once the food

source assigned to the employed bees is drained, the employed bee becomes a scout bee and the food source is abandoned [18].

Each employed bee generates the candidate solution about its source using the following equation.

$$V_i = X_i + \phi_i . (X_i - X_j) \quad (5)$$

Here, Xi is the position of the food source for the ith employed bee, Xj is the food source of a randomly selected employed bee such that $i \neq j$, and ϕ_i is random number between (0, 1).

2.4.2. Onlooker Bees

When the employed bees return to the hive, the onlooker bees are tasked with evaluating their dance performances in order to identify the best source so far based on the following equation.

$$P_i = \frac{fit_i}{\sum_{j=1}^{n} fit_j} \quad (6)$$

Where, fit_i is the value of the fitness function of the i^{th} food source.

The onlooker bees then travel to the best source and evaluate the nectar of its neighbors [17]. The onlooker bees decides which sources are to be abandoned based on the variations in their nectar amounts over the past n cycles.

2.4.3. Scout Bees

The scout bees are tasked with the replacement of abandoned sources with new randomly generated food sources [18]. Once the scout bees generate a new food source, they are employed and are tasked with nectar collection of the source discovered by them.

This process continues for itr number of cycles, and the best source after every cycle is recorded. The best source after the itr^{th} cycle is selected as the optimum solution for the entire problem.

Artificial Bee Colony Algorithm

1. *Initialize the population of solutions x_i*
2. ***while c < itr***
   a. *Generate a candidate solution V_i for the employed bee using equation (5)*
   b. *Apply greedy selection process to choose the best solution for each employed bee*
   c. *Calculate the probability P_i for each solution i using the equation (6)*
   d. *Select best solutions based of the P_i*
   e. *Generate V_i for the selected X_i and evaluate them*

f. *Apply greedy selection process to choose the best solution for each onlooker bee*
g. *Determine the abandoned solutions for the scouts*
h. *Replace the abandoned solutions with randomly generated solutions*
i. *Save the best solution so far.*

end while

3. *Use the final results*

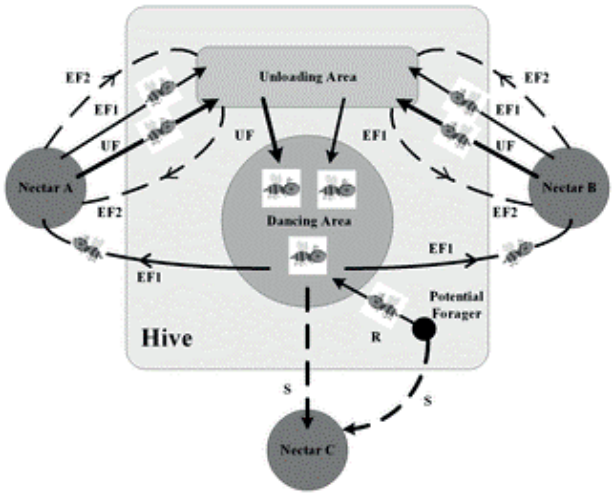

Figure 4. Artificial Bee Colony

2.5. Bat Algorithm

Bat algorithm is based on the capability of the bats to identify prey and other objects based on sonar, known as echolocation [26]. Micro bats emit a very loud sound pulse and hear it when it bounces back from the surrounding objects in order to recognize their surroundings. The signal bandwidth and frequency may vary from specie to specie and is at times a function of the hunting strategies of the bats [26].

The bats adjust their loudness and pulse rate depending on their distance from the prey. The pulses are louder and the pulse rate is lower when the bats are in search of prey. The loudness decreases and the pulse rate increases when the bats began homing towards the prey.

We assume that the bats identify the difference between prey and obstacles and only change their behavior when they reach near their food [26]. The bats are believed to be at a position X_i, flying with a velocity V_i, with a frequency ranging between f_{min} and f_{max}, loudness A_0 and the rate of emission r. The position and velocity of a virtual bat can be calculated using the following equation

$$f_i = f_{min} + (f_{max} - f_{min})\beta \quad (7)$$

$$V_i = V_i + (x_i - x_{best}) f_i \quad (8)$$

$$X_i = X_i + V_i \quad (9)$$

Here, β is a random number between (0,1). We assume that the loudness varies from A_0 to A_{min} as the bat moves towards the prey, and the rate of emission increases simultaneously. We assume that $A_0 = 1$ and $A_{min} = 0$ and that the maximum value of r is r^0. The following equation can be used to vary A and r as the bat moves closer to its prey

$$A_i = \alpha A_i \quad (10)$$

$$r_i = r_i^0 (1 - e^{-\lambda t}) \quad (11)$$

Here, α and λ are constants and are both equal to 0.9 for our calculations. 't' is the number of iterations. It is evident that

$$A_i \to 0 \text{ and } r_i \to r_i^0 \text{ , as } t \to \infty \quad (12)$$

The bat algorithm is run for itr iterations. After every iteration, the bats are ranked on the basis of their distance from the prey. The bat closest to the prey is selected as the best solution for that iteration. After the itrth iteration the best solution is stored as the optimum solution for the entire system and is then used for further calculations.

Figure 5. Eco-location of Micro Bats

Bat Algorithm

1. *Initialize the population of bats x_i and V_i*
2. *Define the pulse frequency f_i for x_i*
3. *Initialize maximum loudness and pulse rates A_0 and r_0*
4. ***while t < itr***
   a. *Generate a new solution by adjusting frequencies and altering the velocities accordingly using equations (7), (8) and (9)*
   b. ***if (rand > r_i)***

Select a solution from the best solutions
Generate a local solution neighboring the selected best solutions
end if

c. *Generate a new random solution*
d. ***if(rand < A_i && cost(X_i) < cost(X_{best}))***
Accept the newly generated solution
Alter A_i and r_i using equations (10) and (11)
end if
e. *Rank the bats and select the globally best solution*

end while

5. *Use the final results*

3. Proposed Work

3.1. Quantum Optimization

The basic building block of a quantum computer is a quantum-bit, which exists in an overlapping state of 0s and 1s. The value of a q-bit is defined as the probability of both these states [28]. The following equations represents a quantum bit where α is the probability of state 1 and β is the probability of state 0.

$$|\psi\rangle = \alpha|1\rangle + \beta|0\rangle \quad (13)$$

$$|\alpha|^2 + |\beta|^2 = 1 \quad (14)$$

A quantum-inspired algorithm uses a complex number to represent a quantum bit where $\alpha = \cos\theta$ and $\beta = \sin\theta$ such that the complex number $e^{i\theta}$ represents the quantum bit. The real part of the said complex number denotes the probability of 1 and the complex part denotes the probability of 0.

The quantum Optimization Algorithm uses the normalized form of all the inputs instead of their actual values [28]. This allows the user to store these inputs in the form of complex numbers with magnitude one. Each complex number is equipped to store two such values and can be used to generate quantum-inspired counterparts of already existing algorithms.

Quantum-inspired Particle Swarm Optimization Algorithm

1. *Initialize the bird population ϕ_i (position) with randomly generated arguments between 0 and 2π*
2. *Set the values for the constants ω, c_0, c_1*
3. *Initialize the values of $\Delta\phi_i$ (velocity), $p^{(i)}_{best}$ and g_{best}*
4. ***while t < itr***
   a. *For each particle i, ensure that the particles are in the range (a_i, b_i) using the following equation.*

$$(X_i)_j = 1/2(b_i \times (1+e^{\Phi_{ij}i}) + a_i((1-e^{\Phi_{ij}i})))$$

b. *Use the imaginary and real parts of the solution to generate two*Δ
c. *different solutions.*
d. *Calculate the cost of all the solutions*
e. *Use greedy approach to choose the best solution for the particle considering both real and imaginary solutions.*
f. *Find the particle with the minimum cost.*
g. ***If (cost(P_i) $_{(min)}$ < g_{best})***
$g_{best} = P_{i\,(min)}$
end if
h. *Calculate the velocity and next position using the following equations*

$$(\Delta\Phi_{ji})_{op} = \begin{bmatrix} 2\pi + (\Phi_{ji})_{op} - \Phi_{ji}, ((\Phi_{ji})_{op} - \Phi_{ji} <= -\pi) \\ (\Phi_{ji})_{op} - \Phi_{ji}, (-\pi < (\Phi_{ji})_{op} - \Phi_{ji} < \pi) \\ (\Phi_{ji})_{op} - \Phi_{ji} - 2\pi, ((\Phi_{ji})_{op} - \Phi_{ji} > \pi) \end{bmatrix}$$

$$(\Delta\Phi_{ji})_{g} = \begin{bmatrix} 2\pi + (\Phi_{ji})_{g} - \Phi_{ji}, ((\Phi_{ji})_{g} - \Phi_{ji} <= -\pi) \\ (\Phi_{ji})_{g} - \Phi_{ji}, (-\pi < (\Phi_{ji})_{g} - \Phi_{ji} < \pi) \\ (\Phi_{ji})_{g} - \Phi_{ji} - 2\pi, ((\Phi_{ji})_{g} - \Phi_{ji} > \pi) \end{bmatrix}$$

$$\Delta\Phi_{ji}(t+1) = \omega\Delta\Phi_{ji}(t) + c_1 r_1 (\Delta\Phi_{ji})_{op} + c_2 r_2 (\Delta\Phi_{ji})_{g}$$

$$\phi_{ji}(t+1) = \phi_{ji}(t) + \Delta\phi_{ji}(t+1)$$

i. *Interchange the real and imaginary parts of the position by replacing* ϕ *with* $\frac{\pi}{2} - \phi$

end while

6. *Use the final Results*

Quantum-inspired Artificial Bee Colony Algorithm

1. *Initialize the population of sources* ϕ_i *(position) with randomly generated arguments between 0 and* 2π
2. ***while c < itr***
   a. *For each source i, insure that it lies within the range (a_i, b_i) using equation (15)*
   b. *Generate two candidate solution V_i for the employed bee using equation (5), one for both real and imaginary parts of the sources*
   c. *Apply greedy selection process to choose the best solution for each employed bee*
   d. *Calculate the probability P_i for each solution i using the equation (6)*
   e. *Select best solutions based of the P_i*
   f. *Generate V_i for the selected X_i and evaluate them*

g. *Apply greedy selection process to choose the best solution for each onlooker bee*
h. *Determine the abandoned solutions for the scouts*
i. *Replace the abandoned solutions with randomly generated solutions*
j. *Save the best solution so far.*

end while

4. *Use the final results*

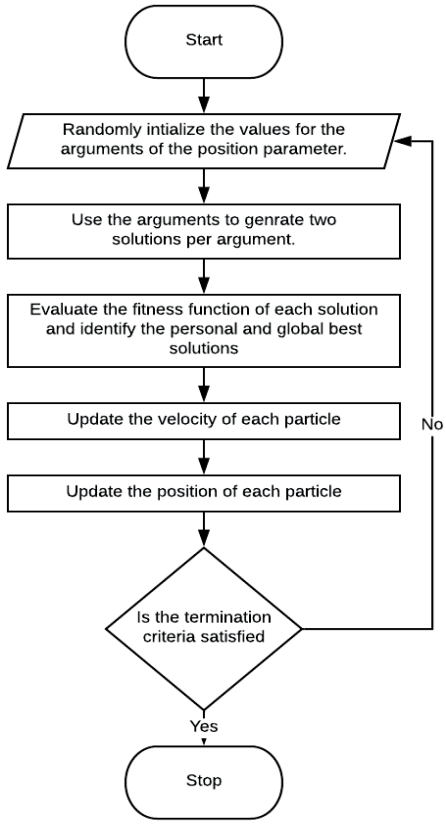

Figure 6. Flow of control for qPSO

Quantum-inspired Bat Algorithm

1. *Initialize the population of bats X_i ϕ_i (position) with randomly generated arguments between 0 and 2π*
2. *Initialize V_i*
3. *Define the pulse frequency f_i for X_i*
4. *Initialize maximum loudness and pulse rates A_0 and r_0*
5. ***while t < itr***
   a. *Generate a new solution by adjusting frequencies and altering the velocities accordingly using equations (7), (8) and (9)*

b. *For each bat i, insure that it lies within the range (a_i, b_i) using equation (15)*
c. *Use the real and imaginary parts of the bat's position to generate a two simultaneous solutions*
d. ***if (rand > r_i)***
   Select a solution from the best solutions
   Generate a local solution neighboring the selected best solutions
   end if
e. *Generate a new random solution*
f. ***if(rand < A_i && cost(X_i) < cost(X_{best}))***
   Accept the newly generated solution
   Alter A_i and r_i using equations (10) and (11)
   end if
g. *Rank the bats and select the globally best solution*

end while

6. *Use the final results*

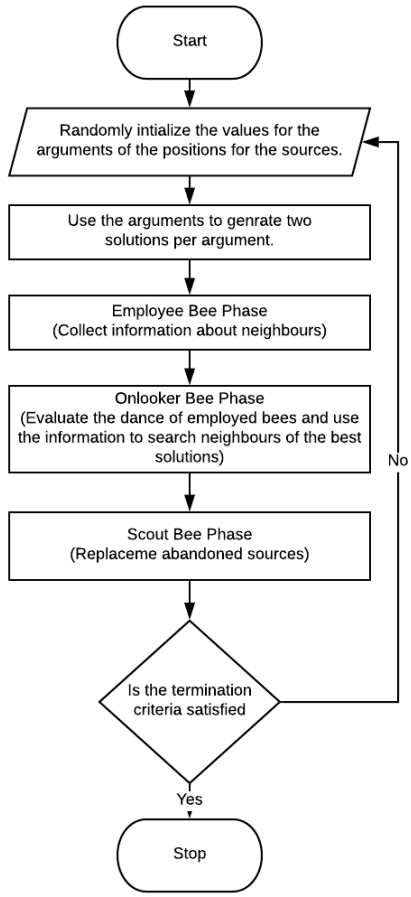

Figure 7. Flow of control for qABC

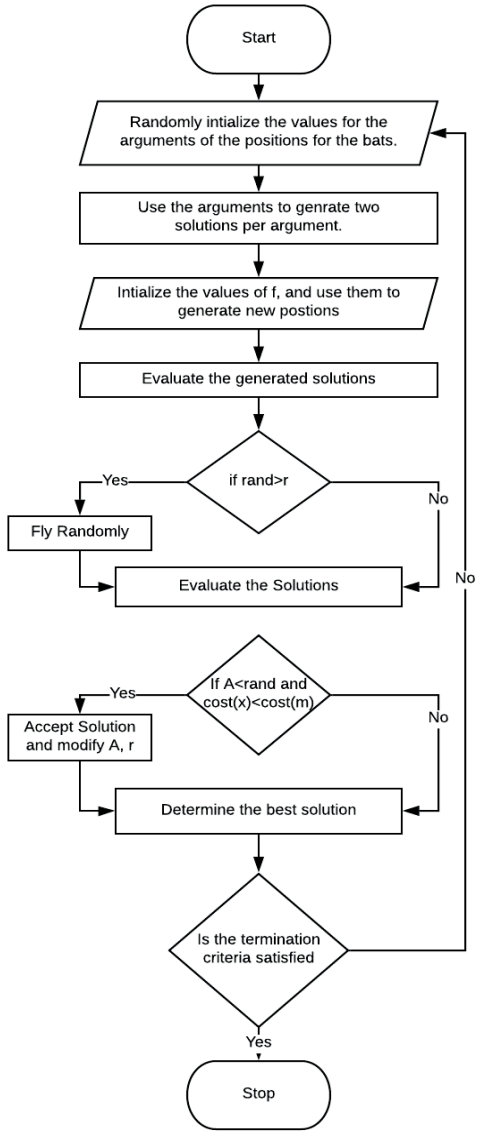

Figure 8. Flow of control for qBat

4. Results

The proposed model was fed a subset of 16 features from the given set of 23 features which were carefully selected after feature selection. These proposed model was then trained using PSO, ABC, Bat Algorithm, qPSO, qABC, and qBat. All of these algorithms were run 10,000 times with a population size of 1000. It was observed that qPSO was the fastest among the given algorithms whereas, qABC was the slowest. It was also observed that qABC provided the highest accuracy of about 92.3% whereas, PSO was the least accurate algorithm with an accuracy of about 86.48%. These results

were observed with a data set of only 195 data points which is not enough for accurate training. Hence, the efficiency of these results can be increased substantially with the use of a larger data set.

The following figures display the variations of the cost function of these algorithms with respect to the number of iterations and also draws comparisons between their performances in terms of time used per iteration and the accuracy of the final predictions.

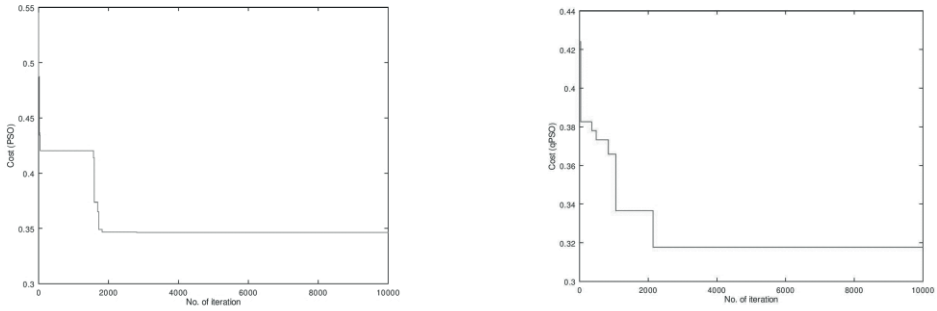

Figure 9. Variations of the cost function of the PSO and qPSO based Neural Network Classification models

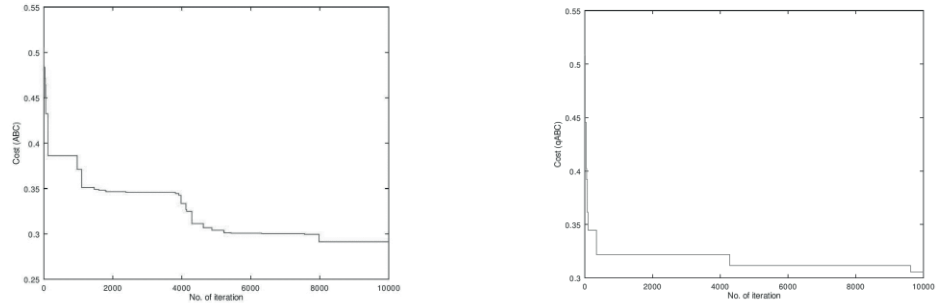

Figure 10. Variations of the cost function of the ABC and qABC based Neural Network Classification models

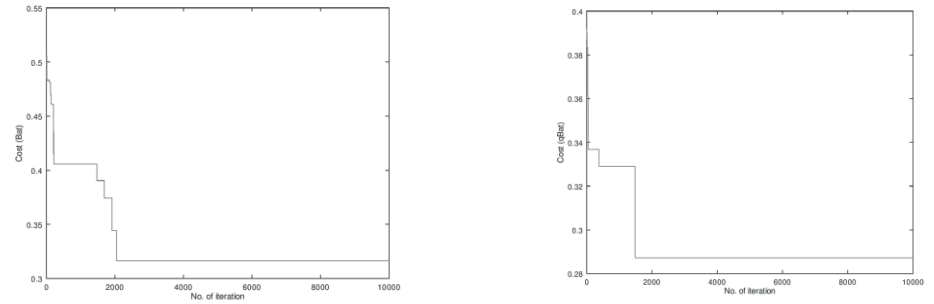

Figure 11. Variations of the cost function of the Bat and qBat based Neural Network Classification models

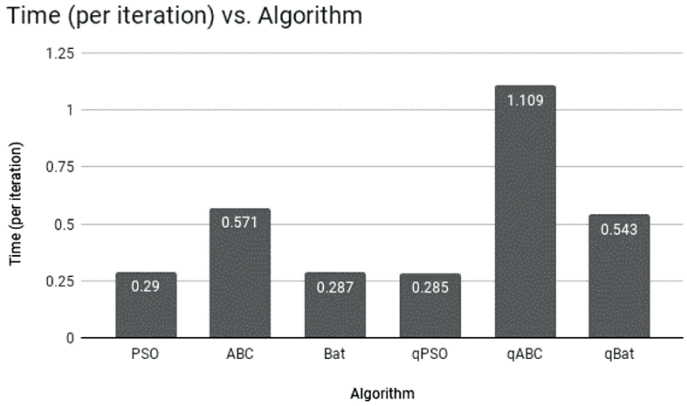

Figure 12. Time utilized per unit iteration

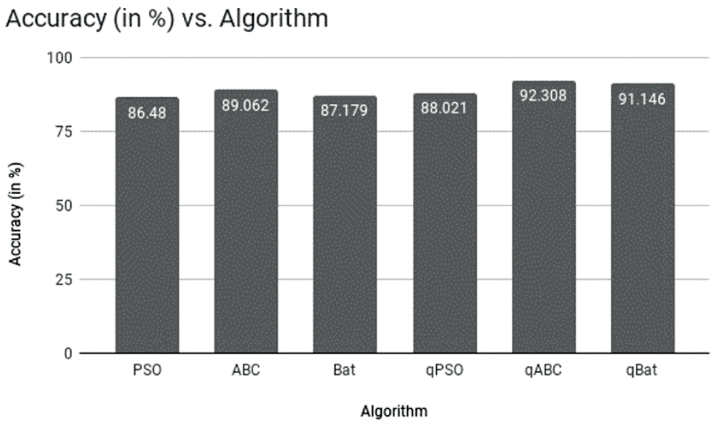

Figure 13. Accuracy of the Model

5. Conclusion

Optimization based evolutionary algorithms such as PSO, ABC and Bat Algorithms are simple yet effective algorithms. These algorithms can be used for any kind of optimization problem including feature selection, classification and even neural network weight distribution. It was observed that among the three algorithms, ABC showed best results for neural network weight distribution. It was also observed that quantum optimization increases the accuracy of these models to some extent and offers a more accurate as well as space-time efficient result. The paper concludes that qABC (quantum-inspired Artificial Bee Colony) is the most powerful of the applied algorithms for neural network weight distribution and can be used not only in disease diagnosis or neural network based classification models but can find its applications in various other neural network based training models.

References

[1] Parkinson's Disease Foundation: http://www.parkinson.org/understanding-parkinsons/10-early-warning-signs.

[2] Sahni S., Aggarwal V., Khanna A., Gupta D., Bhattacharyya S.: "Diagnosis of Parkinson's Disease using a Neural Network based on qPSO". ICICC - International Conference on Innovative Computing and Communication (2019).

[3] Little, M. A., McSharry, P. E., Roberts, S. J., Costello, D. A., & Moroz, I. M.: "Exploiting Nonlinear recurrence and Fractal scaling properties for voice disorder detection". Biomedical Engineering Online, 6 (2007).

[4] Thomas Back: "Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms". Oxford University Press, Inc. New York, NY, USA (1996).

[5] Akama S: "Elements of quantum computing". Springer, Berlin (2015).

[6] M. A. Nielsen and I. L. Chung: "Quantum Computation and Quantum Information", Cambridge University Press, Cambridge UK, (2000).

[7] Pittenger AO.: "An introduction to quantum computing algorithms", Birkhäuser; (2000).

[8] Y. Huang and S. Wang: "Multilevel Thresholding Methods for Image Segmentation with Otsu Based on QPSO," 2008 Congress on Image and Signal Processing, Sanya, Hainan, pp. 701-705 (2008).

[9] O. Vinyals and S. V. Ravuri, "Comparing multilayer perceptron to deep belief network tandem features for robust ASR," in Proc. ICASSP, 2011, pp. 4596–4599.

[10] Q. V. Le, J. Ngiam, A. Coates, A. Lahiri, B. Prochnow, and A. Y. Ng, "On optimization methods for deep learning," in Proc. 28th Int. Conf. Machine Learning, 2011, pp. 265–272.

[11] C. Plahl, T. N. Sainath, B. Ramabhadran, and D. Nahamoo, "Improved pretraining of deep belief networks using sparse encoding symmetric machines," in Proc. ICASSP, 2012, pp. 4165–4168.

[12] B. Hutchinson, L. Deng, and D. Yu, "A deep architecture with bilinear modeling of hidden representations: Applications to phonetic recognition," in Proc. ICASSP, 2012, pp. 4805–4808.

[13] Cramer, J.S., "The Origins of Logistic Regression". Tinbergen Institute Working Paper No. 2002-119/4. Available at SSRN: https://ssrn.com/abstract=360300 or http://dx.doi.org/10.2139/ssrn.360300 (December 2002).

[14] Berkson J., "Application of Logistic Function to bio-assay". Journal of the American Statistical Association, Vol. 9, 357-365 (1994).

[15] Kennedy J.: "Particle Swarm Optimization". In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston, MA (2011).

[16] J. Kennedy and R. Eberhart: "Particle swarm optimization", Proceedings of ICNN'95 - International Conference on Neural Networks, Perth, WA, Australia, pp. 1942-1948 vol.4. (1995).

[17] Karaboga, D. and Akay, B. "Artificial bee colony (ABC) algorithm on training artificial neural networks", in Signal Processing and Communications Applications, SIU 2007, IEEE 15th, IEEE, pp.1–4. (2007)

[18] Karaboga, D. and Akay, B. "A comparative study of artificial bee colony algorithm", Applied Mathematics and Computation, Vol. 214, No. 1, pp.108–132. (2009)

[19] Irani, R. and Nasimi, R. "Application of artificial bee colony-based neural network in bottom hole pressure prediction in underbalanced drilling", Journal of Petroleum Science and Engineering, Vol. 78, No. 1, pp.6–12. (2011)

[20] Jiao, J., Yao, S. and Xia, C. "Application for artificial bee colony algorithm in migration of mobile agent", in Advanced Intelligent Computing, Theories and Applications: 6th International Conference on Intelligent Computing, Changsha, China, August 18–21, Springer-Verlag, New York Inc., Vol. 93, p.232. (2010)

[21] Krohling, R. A. Gaussian Swarm. "A novel particle swarm optimization algorithm". *Proceedings of the 2004 IEEE conference on cybernetics and intelligent systems* (vol. 1, pp. 372–376). (2004).

[22] Clerc, M., & Kennedy, J.. "The particle swarm: Explosion, stability, and convergence in a multi-dimensional complex space". *IEEE Transactions on Evolutionary Computation, 6*, 58–73. (2002)

[23] Altringham, J. D.: Bats: "Biology and Behaviour", Oxford Univesity Press, (1996).

[24] Goldberg, D.: "Genetic Algorithms in Search, Optimization and Machine Learning". Addison-Wesley, Reading, MA, (1989).

[25] Kennedy, J. and Eberhart, R., Swarm Intelligence. Academic Press, (2001).

[26] Yang, Xin-She: "A New Metaheuristic Bat-Inspired Algorithm". 284. 10.1007/978-3-642-12538-6_6. (2010)

[27] X.-S. Yang, Nature-Inspired Metaheuristic Algorithms, Luniver Press, (2010)

[28] Kuk-Hyun Han and Jong-Hwan Kim: "Quantum-inspired evolutionary algorithm for a class of combinatorial optimization," in IEEE Transactions on Evolutionary Computation, vol. 6, no. 6, pp. 580-593 (2002).

[29] Provost, F. & Kohavi, R.: "Machine Learning", Kluwer Academic Publishers, Boston pp. 30-127 (1998).

[30] Gupta, D., Sundaram, S., Khanna, A., Hassanien, A. E., & de Albuquerque, V. H. C. : "Improved diagnosis of Parkinson's disease based on Optimized Crow Search Algorithm" Computers and Electrical Engineering, 68, pp. 412–424. (2018).

[31] Gupta D, Julka A, Jain S, Aggarwal T, Khanna A, Albuquerque VHCD: "Optimized cuttlefish algorithm for diagnosis of Parkinson's disease", Cognit Syst Res 52. pp: 36–48 (2018).

[32] Gupta D., Rodrigues J. J. P. C., Sundaram S., Khanna A., Korotaev V., Albuquerque V. H. C: "Usability Feature Extraction Using Modified Crow Search Algorithm: A Novel Approach", Neural Computing and Applications (2018).

[33] Aileen K. Ho, Robert Iansek, Caterina Marigliani, John L. Bradshaw, and Sandra Gates: "Speech Impairment in a Large Sample of Patients with Parkinson's Disease", Behavioural Neurology, vol. 11, no. 3, pp. 131-137 (1999).

[34] Dominique Twelves, Kate S.M. Perkins, MCRP , and Carl Counsell: "Systematic Review of Incidence Studies of Parkinson's Disease", Movement Disorders Vol. 18, No. 1, pp. 19 – 31 (2003).

[35] Prerna Sharma, Shirsh Sundaram, Moolchand Sharma, Arun Sharma, Deepak Gupta. "Diagnosis of Parkinson's disease using modified grey wolf optimization", Cognitive Systems Research Vol. 52, pp. 36-48 (2018).

[36] Aditya Khamparia, Aman Singh, Divya Anand, Deepak Gupta, Ashish Khanna, Arun Kumar N, Joseph Tan, "A Novel deep learning based multi-model ensemble methods for prediction of neuromuscular disorders", Neural Computing and Applications (2018).

[37] Jain Saksham, Agrawal Shreya, Paruthi Arpit, Trivedi Ayush, Soni Umang, "Neural Networks for Mobile Data Usage Prediction in Singapore", International Conference on Innovative Computing and Communications, Vol. 2, pp.349-357 (2019).

[38] Radha N, Shahina A, Nayeemulla Khan A, "Improving Recognition of Speech System Using Multimodal" Approach: Proceedings of ICICC 2018, Vol. 2, pp.397-410 (2019).