User Localization in an Indoor Environment Using Fuzzy Hybrid of Particle Swarm Optimization & Gravitational Search Algorithm with Neural Networks

Jayant G. Rohra[1], Boominathan Perumal[1(✉)], Swathi Jamjala Narayanan[1], Priya Thakur[1], and Rajen B. Bhatt[2]

[1] VIT Univerity, Vellore 632014, India
jayant.rohra95@gmail.com, swathi.jns@gmail.com, boominathan.p@vit.ac.in, pthakur93@yahoo.com
[2] Robert Bosch Research Labs, Pittsburgh, USA
rajen.bhatt@gmail.com

Abstract. Detecting users in an indoor environment based on Wi-Fi signal strength has a wide domain of applications. This can be used for objectives like locating users in smart home systems, locating criminals in bounded regions, obtaining the count of users on an access point etc. The paper develops an optimized model that could be deployed in monitoring and tracking devices used for locating users based on the Wi-Fi signal strength they receive in their personal devices. Here, we procure data of signal strengths from various routers, map them to the user's location and consider this mapping as a classification problem. We train a neural network using the weights obtained by the proposed fuzzy hybrid of Particle Swarm Optimization & Gravitational Search Algorithm (FPSOGSA), an optimization strategy that results in better accuracy of the model.

Keywords: Neural networks · Optimization methods · PSO · GSA · PSOGSA · User localization · Wi-Fi signal strength · Fuzzy logic

1 Introduction

Advancements in location based services have enabled wide prospects in mobile computing. Many strategies have been adopted to provide users with custom locality based services. These strategies have shown a tremendous boom in e-commerce revenues, embedded smart systems, location based recommender systems and various other fields. Technologies like the GPS, Bluetooth and Wi-Fi could be exploited to provide such services. Bulusu et al. [1] used GPS methods for user localization, but these methods were used to achieve precision only in certain ranges and cannot be applied to indoor locations due to weak satellite signals. Bluetooth is another technology that can be used to serve this purpose, but it can only be well applied for short ranges. Thus, the user localization by using the Wi-Fi access points could be a better approach. Salazar et al. [2] introduced methods to predict the behavior of people by

K. Deep et al. (eds.), *Proceedings of Sixth International Conference on Soft Computing for Problem Solving*, Advances in Intelligent Systems and Computing 546, DOI 10.1007/978-981-10-3322-3_27

monitoring their daily movements. Such location detection systems could also be used in panic situations and disasters, when people require necessary rehabilitation. Nguyen et al. [3] suggests recognition techniques for patients suffering from severe brain injuries who could be monitored by observing patterns in their movements. Pei et al. [4] proposed SVM techniques and showed better classification rates compared to other existing learning techniques. Cho [5] proposed learning methodologies to categorize the locality of indoor and outdoor regions using the location service logs of smart phones. Zou et al. [6] introduced an indoor localization mechanism based on extreme machine learning strategies and depicted its easy adaptation to versatile environments.

Zadeh [7] introduced the fuzzy set theory that has been widely adopted in many domains like real-time controllers, diagnostic systems etc. Real world data has various dimensions, much more than the classical logic of true or false. The fuzzy logic is used to correlate real life scenarios representing probabilities measuring the degree of truth in the range 0 to 1. Jang and Sun [8] proposed the interesting concept of modelling neural networks with fuzzy logic and parametrizing control. The neuro-fuzzy strategy alone would not be sufficient to attain the best throughput to the neural networks. The need for our problem lies to detect users at right locations using better learning techniques. But most of the techniques proposed lack the apt usage of optimization strategies that train the model rightly. We look into metaheuristic techniques that promise sufficiently good solutions to optimization strategies. Eberhart and Kennedy [9] introduced the Particle Swarm Optimization (PSO) strategy that considers a population of candidate solutions or particles moving around the search space and are updated to their *localBest* or *globalBest* computed using their position and velocity parameters. The standard PSO algorithm suffers from major problems like the ability to explore new search spaces. Shi and Russell [10] introduced an adaptive PSO approach that uses multiple benchmark functions to test the fuzzy system applied in various dimensions of the PSO. Liu and Abraham [11] proposed a fuzzy PSO that highlights the need to explore new search spaces by introducing a turbulence factor in the velocity component of the PSO. However, these algorithms lack the assurance of obtaining the global optimum. Mirjalili et al. [12] overcame this problem by proposing a hybrid, PSOGSA that introduces the ability of the GSA to escape the local optimum and hence improve the accuracy of the neural network. But, this algorithm lacks consistency and saturates at the lower iterations of the search, when the dimensions of the problem are increased. Nandy et al. [13] proposed a bee colony based back propagation approach to train ANN. These techniques thus improvise the fact that MLP based classifiers when trained with optimization approaches do give good performance accuracies. Kawam et al. [14] used the cuckoo swarm and PSO technique to train a MLP and hence depicted the need of using optimization strategies indeed enhances the performance of the neural network considerably. But, various such techniques adopted often lack proper convergence and guarantee that the complete population is explored.

Considering these factors, we propose the FPSOGSA that overcomes the possibilities of trapping itself in the local minima and enhances the probability of a higher convergence rate. At higher iterations, it gradually decreases the error rate rather than attaining saturation, as seen in PSOGSA. It obtains better convergence, enhances the ability of optimizing the neural network and hence reducing the mean square error of the Fuzzy Neural Network (FNN). Many such approaches have been used to train

various classifiers. Artificial Neural Network (ANN) is a model easy to understand and use. More importantly it is nonlinear and non-parametric in nature. ANN is largely used to solve various classification and forecasting problems with the Back Propagation (BP) algorithm. However, the BP convergence is slow and not guaranteed. Therefore, we need to use optimization strategies to attain faster convergence and higher accuracy rates. Hence, we introduce the hybrid PSOGSA strategy as an optimization strategy here. On the other hand, the ANN is said to be a black box learning approach. It cannot deal with uncertainties. To overcome this, we introduce the fuzzy component. Fuzzy is quite good in handling uncertainties and can also interpret the relationship between the input and output by producing rules. Hence, we introduce the FPSOGSA algorithm.

2 User Localization as a Classification Problem

To predict the user's location accurately, a definite and consistent model has to be trained and deployed in a tracking or monitoring device. We measure the Wi-Fi signal strength received from various routers in a bounded location and train the neural network so that it could further predict the user's location for an unknown tuple set having signal strengths. Here, we consider a setup at an office location in Pittsburgh, USA. The office has seven Wi-Fi routers and its signal strengths received from these routers categorize the location of user in the conference room, kitchen or the indoor sports room. Sample data tabulated is shown in Table 1. WS1 corresponds to the signal strength received from the router 1, WS2 corresponds to the signal strength received from the router 2, and similarly for the other routers. The class labels corresponding to the conference room, kitchen and the indoor sport are labelled 1, 2 and 3 respectively. In our setup facility, we have considered an Android device and tabulated strengths of wireless signals captured by the device. At certain locations, the signal strengths were observed by polling the wireless signal strength at a constant time interval (every 1 s considered here). This was again repeated for other locations and suitable data was collected for one thousand and five hundred observations made at this facility for seven different routers. The model developed here, could hence be reused according to the scenario of the bounded location and the number of wireless routers in the physical facility. This data is being formulated into a pattern classification dataset by considering the seven wireless routers as the input dimensions which are used to predict the user's location in an office as one of the three dimensional categories. After having a concrete dataset ready, we now train the neural network using a metaheuristic approach that enhances the chances of classifying the right class label optimally. We discuss our approach of training the model using Fuzzy PSO GSA (FPSOGSA) in Sect. 3.

Table 1. Sample Data for user localization using wireless signal strength

WS1	WS2	WS3	WS4	WS5	WS6	WS7	Class
−64	−56	−61	−66	−71	−82	−81	1
−68	−57	−61	−65	−71	−85	−85	1
−17	−66	−61	−37	−68	−75	−77	2
−16	−70	−58	−14	−73	−71	−80	2
−52	−48	−56	−53	−62	−78	−81	3
-49	-55	-51	-49	-63	-81	-73	3

3 Evolution from the Conventional PSOGSA to FPSOGSA and Training the Neural Network with the Proposed Fuzzy-PSOGSA Algorithm

Mirjalili et al. [12] proposed the PSOGSA by introducing an exploitation capability to the standard PSO algorithm that increases the probability of finding the *globalBest* solution. The novel idea of using mass interactions among particles by including the gravitational search capability, proposed by Rashedi et al. [15] further enhanced the accuracy rates of the FNN. Later in this section, we introduce fuzzy decision parameters of the PSOGSA that decide the need for further exploration of the particle in the search space. Suitable thresholds are set to decide if the particle needs to explore further dimensions. This algorithm would hence fit the need of not missing out on the globalBest, as it gives more exploration ability to the particles.

We initially consider a space with 'N' particles that have randomly allocated positions that are referred to as the current positions (CurrPos) of the particles. The positions of each of these particles have "d" dimensions and a configuration of these positions is considered to be a candidate solution. The forces between the particles in each iteration, are calculated as,

$$F_{ij}^{d}(t) = G(t)\frac{M_{pi}(t) \times M_{aj}(t)}{R_{ij}(t)}\left(CurrPos_{j}^{d}(t) - CurrPos_{i}^{d}(t)\right) \tag{1}$$

where M_{pi} and M_{aj} are passive and active gravitational masses of particles i and j respectively and R_{ij} is computed as the Euclidean distance between the two particles. The total force acting on any particle i is computed as the sum of the forces acting on every other particle in the space. The time variant gravitational constant, $G(t)$ is computed as,

$$G(t) = G_0 \times \exp\left(-k \times CurrentIteration/MaxIteration\right) \tag{2}$$

where k is a descending co-efficient and G_0 is the initial gravitational constant value at t. The mass of each particle is related to the fitness value. It is updated at every epoch using the equation,

$$M_i(t) = \frac{CurrFit_i - best}{best - worst} \tag{3}$$

where *best* is the minimum fitness value for a minimization optimization problem and *worst* is the maximum fitness value. The acceleration of the particle is computed as follows:

$$a_i^d(t) = F_i^d(t) / M_i(t) \tag{4}$$

The weight function, W is calculated using:

$$W = W_{\min} - CurrentIteration \times (W_{\max} - W_{\min}) / MaxIteration \tag{5}$$

Here we initialize W_{min} and W_{max} as suitable minimum and maximum inertia weights.

Now, the velocity of the particle is updated by using the equation:

$$\left(Vel_i^d\right)_{t+1} = W \times \left(\exp loreVel_i^d\right)_t + rand() * a_i^d + rand() * (globalBest_j - CurrPos_i^d) \tag{6}$$

where rand is any number between the range [0,1] and the *globalBest* is the best solution obtained so far. The *exploreVel* is computed by using the fuzzy inference mechanism discussed in the section later.

Consider a neural network as shown in Fig. 1 with seven input nodes as the attributes of the user localization dataset and three output nodes as the class labels. The FPSOGSA trains the neural network by using the exploration and exploitation capabilities of the particles in the search space. As the PSO suffers saturation or slow convergence at the ending few iterations, the particles sometimes do not tend to come out of their constrained search space. This means that the mean square error (MSE) does not further decrease and hence there is very little or no change found in the accuracy of the neural network. Thus, in order to provide particles with an ability to explore new search spaces, we provide an extra velocity component, $exploreVel_{ij}$ that is inferred from a Fuzzy Inference System (FIS). This enhances the search capability of the particles by exploring new dimensions in the search space and hence increasing the chances of obtaining a better *globalBest* solution. As discussed earlier, here we update the mass and acceleration of the particles before obtaining the explore velocity from the FIS. This is because the GSA component adds mass interactions that play a vital role in achieving the global optimum and also the fact that the acceleration component is used to update the velocity of the particle in the $(t + 1)^{th}$ iteration. The FIS takes in the Normalized Current Best Fitness Value (*NCBFV*) and the velocity of the particle (Vel_{ij}) as inputs and infers the scaling factor (S_f) and the velocity threshold control parameter (V_{tc}) as the output using the Fuzzy Rules discussed below in this section. The scaling factor, S_f is obtained as a result to prevent the particle from overshooting off its domain while getting extra exploration capability in the search space.

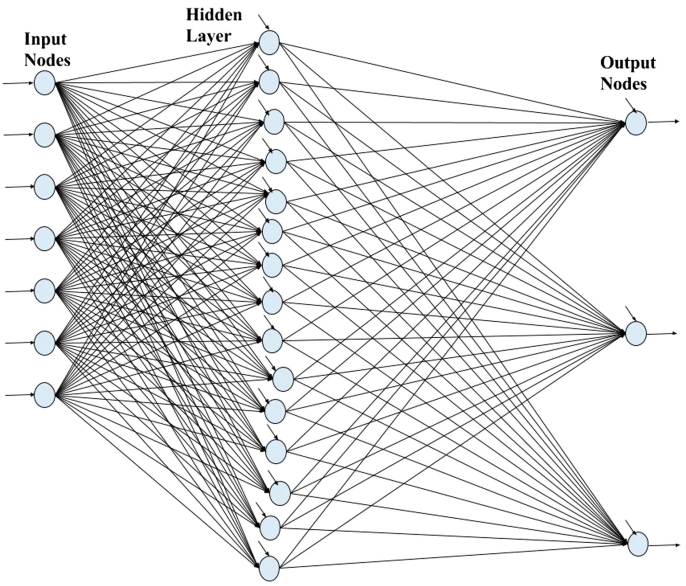

Fig. 1. Neural Network for classification of a dataset with 7 attributes and 3 class labels

The NCBFV is calculated as follows:

$$NCBFV_i = \frac{CurrFit_i - MinFit}{MaxFit - MinFit} \tag{7}$$

where $CurrFit_{i,}$ is the current fitness value of the particle, *MinFit* is the least fitness value obtained by the particle till the current iteration and *MaxFit* is the maximum fitness value obtained.

The threshold (θ) is calculated from the velocity threshold control parameter as follows:

$$\theta = e - [10(1 + V_{tc})] \tag{8}$$

The Vel_{ij} is the latest velocity of the particle attained until the previous iteration. For the first iteration, the $exploreVelocity_{ij}$ is considered to be the same as $Vel_{ij.}$ For iterations after the first, the $exploreVelocity_{ij}$ is obtained by checking for θ as follows:

$$\exp lore\, Velocity_{ij} = \begin{pmatrix} Vel_{ij,} & Vel_{ij} \geq |\theta| \\ UDistb[-1,1] \times \max(Vel_{ij}) / S_f, & Vel_{ij} \leq |\theta| \end{pmatrix} \tag{9}$$

where *UDistb[−1,1]* is an uniform distribution in the range [−1,1], $\max(Vel_{ij.})$ is the maximum value of the velocity obtained till now, θ is a threshold obtained from Eq. (8) and S_f is the scaling factor obtained as one of the results of the FIS. The given fuzzy inference rules are used to obtain the values of the velocity threshold control parameter

(V_{tc}) and the scaling factor (S_f), which determine the $exploreVelocity_{ij}$ of the particle based on the threshold θ.

1. If (*NCBFV* is low) and (*Vel* is low) then (V_{tc} is high)
2. If (*NCBFV* is medium) then (V_{tc} is medium)
3. If (*NCBFV* is high) and (*Vel* is high) then (V_{tc} is low)
4. If (*NCBFV* is low) or (*Vel* is low) then (S_f is large)
5. If (*NCBFV* is medium) then (S_f is medium)
6. If (*NCBFV* is high) or (*Vel* is high) then (Sf is small)
7. If (*Vel* is high) then (V_{tc} is low) (S_f is medium)
8. If (*Vel* is low) then (V_{tc} is high) (S_f is medium)

The weight of each rule is assumed to be one. The fuzzy ranges are chosen suitably, for low/medium/high/small and large depending on the inputs parameters of the variables. Suitable triangular or Gaussian membership functions are used for fuzzification.

Finally, the position of the particle is updated to the next optimal location using:

$$(CurrPos_{ij})_{t+1} = (CurrPos_{ij})_t + (Vel_{ij})_t \tag{10}$$

3.1 Algorithm

1. Begin FPSOGSA
2. *Initialization* – Set a suitable number of iterations as *MaxIteration* to train the FNN.
   a. Initialize the dataset and normalize values in the range [-1, 1].
   b. Select a suitable number of Input, Output and Hidden nodes for the FNN depending on the dataset.
3. Obtain *Weights* to train the Neural Network (NN).
   a. Initialize randomly the weights and bias values.
   b. Choose the number of particles *(N)* and generate the initial population configuration of particles.
   c. Compute the fitness values of each particle and store the best and worst fitness values.
      Computation– Updating and calculating the parameters of the particle in the search space.
   d. Update *G* using the Eq. (2) and compute the *globalBest* for each particle.
   e. Calculate the *mass*, *force* and the *acceleration* of each particle using the Eqs. (3), (1) and (4) respectively.
   f. Update the inertia weights using the Eq. (5).
      Fuzzification – to obtain the *exploreVel*
   g. Obtain and normalize the current best fitness value of the particle using the Eq. (7).

h. Initialize the fuzzy inference system and infer the output variables, S_f and V_T using the rules defined above.
i. Obtain the velocity threshold θ, using the Eq. (8) and compute the *exploreVel* using Eq. (9).
j. Update the velocity of the particle and the new position using the Eqs. (6) and (10) respectively.

4. *Training* – Train the NN by passing the obtained weights
5. Obtain the mean square error of the FNN and compute the classification accuracy of the FNN.
6. Repeat the above process until *CurrentIteration* = *MaxIteration*.
7. End FPSOGSA.

4 Experimental Computational Results and Discussion

The inputs to the network model are the seven attributes of wireless signal strengths measured from the various routers. The outputs obtained are the class labels that classify users based on their locality. 15 hidden nodes are chosen for the neural network structure. The weights for the neural network are obtained from the optimization algorithms. The neural network shown in Fig. 1, is being trained with PSO, GSA, PSOGSA and the proposed FPSOGSA algorithms separately to obtain the initial weights required to train the neural network. These weights are further optimized over 300 iterations to obtain the best accuracy for the dataset. The classification accuracies of the neural networks after training with these algorithms for an evolution of 300 iterations are shown in Table 2. The proposed FPSOGSA boosts the performance of the neural network as it enhances the probability of exploring new search spaces and exploits the best particles so that they overcome the local minima. This is evident from the steep decrease in the Mean Square Error (MSE) values as shown in the Fig. 3. The figure also depicts the comparison in the decrease in mean square error values over three hundred iterations for the various other optimization strategies considered here.

Table 2. Classification Rates in (%)

PSO-NN	GSA-NN	PSOGSA-NN	FPSOGSA-NN	SVM	Naïve Bayes
64.66	77.53	83.28	95.16	92.68	90.47

We can clearly observe that there is very minimal error when the weights are obtained by FPSOGSA to train the neural network. Thus, the FPSOGSA is found to outperform the Particle Swarm Optimization (PSO), Gravitational Search Algorithm (GSA) and the hybrid PSOGSA (Fig. 2).

We also compare other models like the SVM and Naïve Bayes which are commonly used pattern classification approaches. However, we choose to use the neural network due to the concrete reasons explained towards the end of Sect. 1. Here, Fig. 3. shows the classification accuracies obtained using various algorithms for the dataset

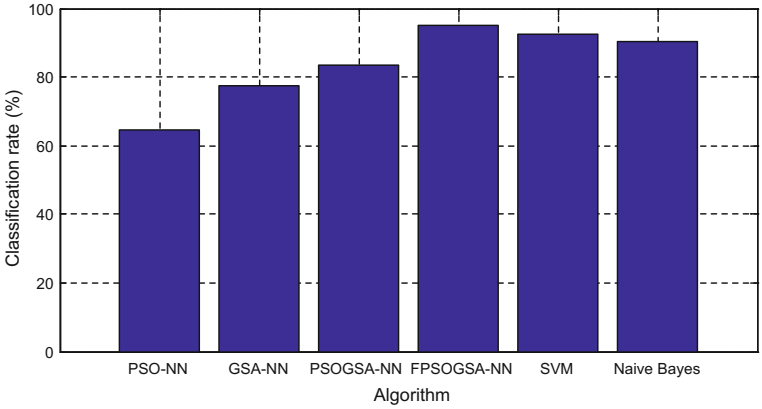

Fig. 2. Comparison of the classification accuracies of various models.

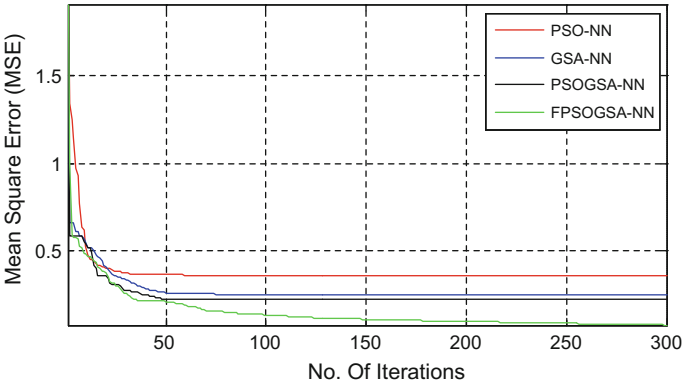

Fig. 3. Plot of MSE for user localization dataset considered for training with PSO, GSA, PSOGSA and FPSOGSA

procured. We performed tenfold cross validation and recorded the average of ten folds as the classification accuracy. From the results obtained we conclude that the FPSOGSA with neural networks give the highest classification rate.

References

1. Bulusu, N., Heidemann, J., Estrin, D.: GPS-less low-cost outdoor localization for very small devices. IEEE Personal Commun. **7**(5), 28–34 (2000)
2. Salazar, A.M., Warden, D.L., Schwab, K., Spector, J., Braverman, S., Walter, J., Ellenbogen, R.G.: Cognitive rehabilitation for traumatic brain injury a randomized trial. JAMA **283**(23), 3075–3081 (2000)

3. Nguyen, N.T., Bui, H.H., Venkatsh, S., West, G.: Recognizing and monitoring high-level behaviors in complex spatial environments. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, II-620. IEEE (2003)
4. Pei, L., Liu, J., Guinness, R., Chen, Y., Kuusniemi, H., Chen, R.: Using LS-SVM based motion recognition for smartphone indoor wireless positioning. Sensors **12**(5), 6155–6175 (2012)
5. Cho, S.B.: Exploiting machine learning Techniques for location recognition and prediction with smartphone Logs. Neurocomputing (2015)
6. Zou, H., Lu, X., Jiang, H., Xie, L.: A fast and precise indoor localization algorithm based on an online sequential extreme learning machine. Sensors **15**(1), 1804–1824 (2015)
7. Zadeh, L.A.: Fuzzy sets. Inform. control **8**(3), 338–353 (1965)
8. Jang, J.S., Sun, C.T.: Neuro-fuzzy modeling and control. Proc. IEEE **83**(3), 378–406 (1995)
9. Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings of the Sixth International Symposium on Micro Machine and Human Science, vol. 1, pp. 39–43 (1995)
10. Shi, Y., Eberhart, R.C.: Fuzzy adaptive particle swarm optimization. In: Proceedings of the 2001 Congress on Evolutionary Computation, vol. 1, pp. 101–106. IEEE (2001)
11. Liu, H., Abraham, A., Zhang, W.: A fuzzy adaptive turbulent particle swarm optimisation. Int. J. Innovative Comput. Appl. **1**(1), 39–47 (2007)
12. Mirjalili, S., Hashim, S.Z.M., Sardroudi, H.M.: Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm. Appl. Math. Comput. **218**(22), 11125–11137 (2012)
13. Nandy, S., Sarkar, P.P., Das, A.: Training a feed-forward neural network with artificial bee colony based backpropagation method. arXiv preprint arXiv:1209.2548 (2012)
14. Kawam, A.A., Mansour, N.: Metaheuristic optimization algorithms for training artificial neural networks. Int. J. Comput. Inf. Technolgy **1**, 156–161 (2012)
15. Rashedi, E., Nezamabadi-Pour, H., Saryazdi, S.: GSA: a gravitational search algorithm. Inform. Sci. **179**(13), 2232–2248 (2009)