ORIGINAL RESEARCH

Performance analysis of neuro swarm optimization algorithm applied on detecting proportion of components in manhole gas mixture

Varun Kumar Ojha, Paramartha Dutta

Department of Computer & System Sciences, Visva-Bharati University, Siksha Bhavana, Santiniketan, India.

Correspondence: Varun Kumar Ojha. Address: Department of Computer & System Sciences, Visva-Bharati University, Siksha Bhavana, Santiniketan 731235, India. Telephone: 91-900-757-1689. E-mail: varun.kumar.ojha@gmail.com

Received: May 20, 2012 **Accepted:** July 2, 2012 **Published:** September 1, 2012
DOI: 10.5430/air.v1n1p31 **URL:**http://dx.doi.org/10.5430/air.v1n1p31

Abstract

The article presents performance analysis of the neuro swarm optimization algorithm applied for the detection of proportion of the component gases found in manhole gas mixture. The hybrid neuro swarm optimization technique is used for implementing an intelligent sensory system for the detection of component gases present in manhole gas mixture. The manhole gas mixture typically contains toxic gases such as Hydrogen Sulfide, Ammonia, Methane, Carbon Dioxide, Nitrogen Oxide, and Carbon Monoxide. A semiconductor based gas sensor array used for sensing the gas components consists of many sensor elements, where each sensor element is responsible for sensing particular gas component. Presence of multiple gas sensors for detecting multiple gases results in cross-sensitivity. The central theme of this article is the performance analysis of the algorithm which offers solution to multiple gas detection issue. The article also presents study on the computational cost incurred by the algorithm.

Keywords

Gas mixture, Gas sensor array, Cross-sensitivity, Particle swarm optimization, Neural network, Optimization, Computational complexity

1 Introduction

The article offers detail study on the implementation and performance analysis of neuro swarm optimization algorithm proposed for the development of an intelligent sensory system for detection of proportion of different component gases present in typical manhole gas mixture. When the domestic and industrial waste products are decomposed into the sewer pipeline network, poisonous gaseous mixture is formed, known as manhole gas mixture. This gaseous mixture usually contains gases like, Hydrogen Sulfide (H_2S), Ammonia (NH_3), Methane (CH_4), Carbon Dioxide (CO_2), Nitrogen Oxide (NO_x), etc [1, 5, 11]. The manholes are built across the sewer pipeline network for cleaning and maintenance purpose. A person as conventional practice has to get down into the pipeline network to serve this purpose. Many pedestrian also becomes vulnerable with these manholes as they are built on the roads or on the road sides. In recent days few instances of death including municipality labourers and pedestrians are reported due to toxic gas exposures [13]. There is also an environmental pollution concern due to exposed manhole gases. The twin problems lead us to mould our research involvements in this direction.

We are using neural network classifier to design an intelligent sensory system for the detection of proportion of component gases present in manhole gas mixture. The neural network has to be trained such that it can act like an intelligent agent who can report the proportional presence of toxic gas components in the manholes whenever it is dropped down into manholes for the detection. In present article the gas detection problem is treated as a pattern recognition problem, where the neural network classifier is trained in supervised mode using the particle swarm optimization algorithm. The training of the neural network is done externally using the particle swarm optimization approach leading to a hybrid approach of neuro swarm optimization algorithm. An intelligent sensory system like this will help labourers to be watchful against the presence of toxic gases while entering into the manholes. In [14, 15, 16, 17, 18, 19] concerned authors presented their respective approaches towards solution to manhole gas detection issue. A semiconductor based gas sensor array containing distinct semiconductor type gas sensors. The sensors in array sense the presence of gases according to concentration in manhole gas mixture. Sensed values by the gas sensor array are cross-sensitive as multiple gases are present in the manhole gas mixture and even if the distinct gas sensors are sensitive to their target gases they are influenced by other gases too. Our objective is to train the neural network such that the cross-sensitivity effect can be minimized and the proposed system can generate flawless report.

In the article, first we discuss the mechanisms in the design of a gas detection system which includes the idea and the complete design of the experimental setup for gas detection. We discuss the process of the formation of data samples used in experimentation. Then we have discussion on the cross-sensitivity issue in this section. Then we discuss and define the training pattern required for the training of neural network. In neuro swarm optimization technique section we elaborately discuss the hybrid concept of neuro swarm optimization algorithm. The performance analysis section is the central theme of our article. In this section we present comprehensive discussion and analysis on the performance of the neuro swarm optimization algorithm. The analysis encompasses tuning different underling parameters. Finally in the result section we pro avide clear picture on the output representation method of the intelligent sensory system.

2 Mechanisms

The section deals with the procedure in the development of an experimental setup for the detection of mixed gases. In this section we present an overview of the gas detection system followed by the data formation technique and definition of training pattern for the neural network classifier.

2.1 Experimental setup

The basic model of intelligent sensory system for detection of manhole gas mixture is shown in Figure 1. In [3, 6, 25] authors have demonstrated their ideas and perspectives toward the design of gas detection system and also have shown their own mechanisms for this purpose.

Our developed gas detection system constitutes of three modules, input module, intelligent module and output module. The input module is an integration of gas chamber, gas sensor array [2, 12] and data preprocessor block. The intelligent module receive data from the input module and after performing the computation on those data it sends its result to the output module for presenting system output in more structured way. The sample of gas mixture is collected into a gas mixture chamber and subsequently allowed to pass over the semiconductor based gas sensor array. The sensor elements in the gas sensor array are responsive to their target gas. Although the gas sensors elements are made to detect their target gases they show sensitivity to other gases also. So, the sensor array response is always involving cross-sensitivity effect [8]. The pre-processing block receives sensed data values from the gas sensor array and these data values are normalized before feeding it to the neural network. The neural network has to be is trained using the normalized data. The output module performs the task of denormalization of the data which is coming out of the neural network. This module also generates alarm if any of the toxic gas components exceeds their safety limit. For the training of the network, several data samples are produced. The section subsequent to this describes the data formation and collection process.

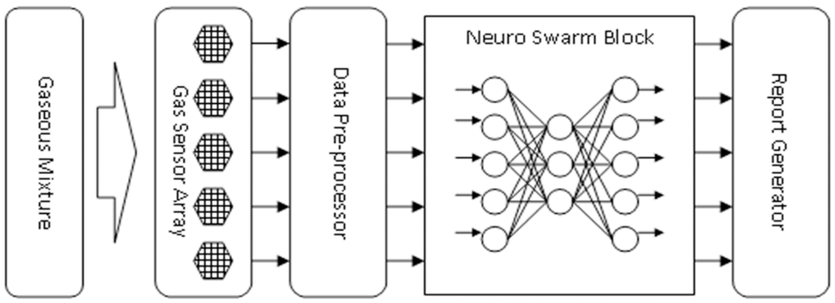

Figure 1. Overview of Gas Detection System.

The figure is showing the flow of the work or the order of the task which may carry out during a typical gas detection process. At first gas mixture chamber collects gases and then allowed it to pass over gas sensor array. The pre-processor bloc provides normalized data value to neural network. The neural network performs its computation on normalized data and sends it to report generator module.

2.2 Data collection

Initial step in data sample formation procedure is the collection of information about the safety limits of the component gases found in manhole gas mixture. Then we prepare several gas mixture samples by mixing gas components in different combination of concentration. The concentrations of the gases in the mixture are taken around their safety limits. The known gas mixture is a synthetic mixture of gases in the known concentration. To prepare a data sample, the mixture of known concentration of gases is blown over the sensor array and the sensor responses are collected in tabular form. In this way we prepare several data samples. A typical example of such data sample is shown in Table 1. Focusing on the first and second row of the Table 1 we can appreciate the inherent cross-sensitivity effect in the responses of sensors. From the first and second sample it is observed that the concentration of only Methane gas is increased and even if there is change in concentration of Methane gas, there is change in responses of all the sensors including the sensor for Methane. It is indicating that the prepared data sample is containing the cross-sensitive effect and it is also observed that this effect is not random; rather it is following some characteristics and pattern. So the sensor responses of the gas sensor array may not be used directly to predict/report the concentration of the gases in manhole gas mixture. To predicted/forecast the concentration of the gases in the manhole gas mixture we need to use intelligent system using pattern recognition techniques.

Table 1. A typical data sample

#	Concentration(PPM) combination					Sensor response				
Sample	NH_3	CO	H_2S	NO_2	CH_4	NH_3	CO	H_2S	NO_2	CH_4
1	50	100	100	100	2000	0.0531	0.0863	0.0733	0.0267	0.1101
2	50	100	100	100	5000	0.0812	0.1122	0.0749	0.0333	0.1934
3	50	100	100	200	2000	0.0963	0.1182	0.0929	0.0577	0.1195
4	50	100	200	200	5000	0.1212	0.12905	0.1291	0.0699	0.2086
5	50	100	200	400	2000	0.1451	0.1495	0.1399	0.0795	0.1207
6	100	200	200	400	5000	0.1569	0.1573	0.1526	0.0799	0.2559
7	100	200	100	200	2000	0.1693	0.1699	0.0722	0.0615	0.1400
8	100	200	100	200	5000	0.1715	0.1798	0.0883	0.0705	0.3000
9	100	200	100	400	2000	0.1821	0.2231	0.0996	0.1302	0.1544
10	100	200	200	400	5000	0.1924	0.2584	0.1869	0.1648	0.3124

Note. The data sample shown here is prepared using mixture of five gases. The table is having three major columns for sample number, for sample gas mixture and for sensor responses respectively. All the samples are mixed in different combination of concentration value of gases shown in second column. The values of sensor response when sample mixtures are blown over gas sensor array in reset condition of sensor array element are shown in third column

2.3 Data pre-processing

Neural network may not forecast accurately with raw data. So the normalization of the data sample is necessary before feeding it to the neural network for processing. By normalization we mean to transform data within range 0 and 1. The standardization of input data for the neural network makes its training faster and reduces the chance of getting stuck to local optima. It also reduces the chance of adversity which may come during the testing of the neural network. The pre-processing block normalizes the data samples according to Equations (1) and (2). According to Equation (1) the NC_{si} (normalized concentration) of gas H_2S of sample 2 is given by 100/5000, where the 100 appearing in the numerator is the C_{si} (concentration of the gas itself) value and the 5000 in the denominator is the C_{max} (maximum concentration among all the samples) value. Similarly, the sensor responses are also normalized according to Equation (2) where, NR_{si} is normalized sensor response, R_{si} is sensors individual response and R_{max} is the maximum response among all the samples.

$$NC_{si} = C_{si} / C_{max} \tag{1}$$

$$NR_{si} = R_{si} / R_{max} \tag{2}$$

The normalized values shown in the Table 2 are used for training of the neural network. Once the network is trained it is ready to be operative in test environment. The output neural network need to be denormalized to report the system output in terms of concentration (parts per million) of gases present in the given test environment. The denormalization is a simple method where the value of network output is multiplied with C_{max} (maximum concentration among all the samples) value. The denormalization method is given in Equation (3)

$$\text{System Output} = \text{Network Output} \times C_{max} \tag{3}$$

Table 2. The normalized data sample

#	Sample mixture					Sensor response				
Sample	NH_3	CO	H_2S	NO_2	CH_4	NH_3	CO	H_2S	NO_2	CH_4
1	0.01	0.02	0.02	0.02	0.4	0.1699	0.2762	0.2346	0.0854	0.3524
2	0.01	0.02	0.02	0.02	1.0	0.2599	0.3591	0.2397	0.1065	0.6190
3	0.01	0.02	0.02	0.04	0.4	0.3082	0.3783	0.2973	0.1846	0.3825
4	0.01	0.02	0.04	0.04	1.0	0.3879	0.4130	0.4132	0.2237	0.6677
5	0.01	0.02	0.04	0.08	0.4	0.4644	0.4785	0.4478	0.2544	0.3863
6	0.02	0.04	0.04	0.08	1.0	0.5022	0.5035	0.4884	0.2557	0.8191
7	0.02	0.04	0.02	0.04	0.4	0.5419	0.5438	0.2311	0.1968	0.4483
8	0.02	0.04	0.02	0.04	1.0	0.5489	0.5755	0.282	0.2256	0.9603
9	0.02	0.04	0.02	0.08	0.4	0.5829	0.7141	0.3188	0.4167	0.4942
10	0.02	0.04	0.04	0.08	1.0	0.6158	0.8271	0.5982	0.5275	1.0000

Note This table is simply contains the normalized value corresponding to the table 1. The table is having three major columns indicating sample number, indicating sample gas mixture and indicating sensor responses respectively. The second column contains the normalized value of the concentration of gas mixture. The third column contains the normalized value of sensor responses.

2.4 Neural network training pattern

The supervised mode of training is used for the training of neural network classifier. So the training pattern comprises of input vector and target vector. It is also mentioned above that the normalized sensor responses are fed as input to the neural network. So the input vector 'I' is created using the normalized values of the sensor responses. In the given data sample input vector is a five element vector. One vector element is for each gas in the sample gas mixture. For sample 1 of Table 2, the input vector ‘I’ can be represented as follows

$$I = [0.1699, 0.2762, 0.2346, 0.0854, 0.3524]$$

The output of intelligent system is presented in terms of the concentration of gases. So the target vector ‘T’ is formed using the gas mixture sample. In the given data sample target vector is a five element vector. One vector element is for each gas in the sample gas mixture. For sample 1 of Table 2, the target vector ‘T’ can be represented as follows

$$T = [0.01, 0.02, 0.02, 0.02, 0.4]$$

From the table 2 it may be observed that the second major column is taken as target vector and third major column is taken as input vector.

3 Neuro swarm optimization technique

It is already discussed that we are inclined to use of neural network classifier for the development of intelligent sensory system. The neural network classifier is trained using particle swarm optimization algorithm. In [7, 21, 22] the corresponding authors demonstrate hybrid PSO based neural network training algorithm. A population based search algorithm is used to search out optimized synaptic weights for a multilayer perceptron. The following subsections offer detailed discussions on the implementation procedure of the neuro swarm optimization algorithm.

3.1 Neuro swarm optimization approach

The neuro swarm optimization approach uses the particle swarm optimization algorithm to search out the optimum combination of the synaptic weights for which the neural network produces minimum sum of squared error for a given input training patterns. Simon [23] defines the neural network as a massively parallel distributed processor combination that has a natural propensity for storing experiential knowledge and making it available for use. The particle swarm optimization algorithm is a population based search algorithm used for the optimization of objective function for given problems. It belongs to the class of stochastic algorithm. The particles of the particle swarm optimization are placed in the search space associated to some problem or function, and evaluate the objective function at its current location. Each particle then determines its movement through the search space by combining some aspect of the history of its own current and best (best-fitness) locations with those of one or more members of the swarm, with some random perturbations. The next iteration takes place after all particles have been moved. Eventually the swarm as a whole, like a flock of birds collectively searching for food, is likely to move close to an optimum of the fitness function [9, 10].

A flow diagram of the neuro swarm optimization algorithm is offered in the Figure 2. The neuro swarm optimization method starts with the formation of neural network and initialization of its synaptic weights. Then the initialization of the particles and parameters of the particle swarm algorithm takes place. The initialization of the parameters of the algorithm is based on the performance analysis shown in the performance analysis section. Thereafter fitness values of the particles are computed. The fitness values of the particles are evaluated by computing the sum of squared error induced by neural network based on the particle location value and given input pattern. A decision block checks whether the fitness value of the global best particle (gBestparicle – described in the section subsequent to this) is good enough to accept. If it is, then algorithm terminates, and gBest particle is displaed as system output else the velocity and position of the particle are updated. The algorithm repeats for all the above procedures until the algorithm encounters max iteration.

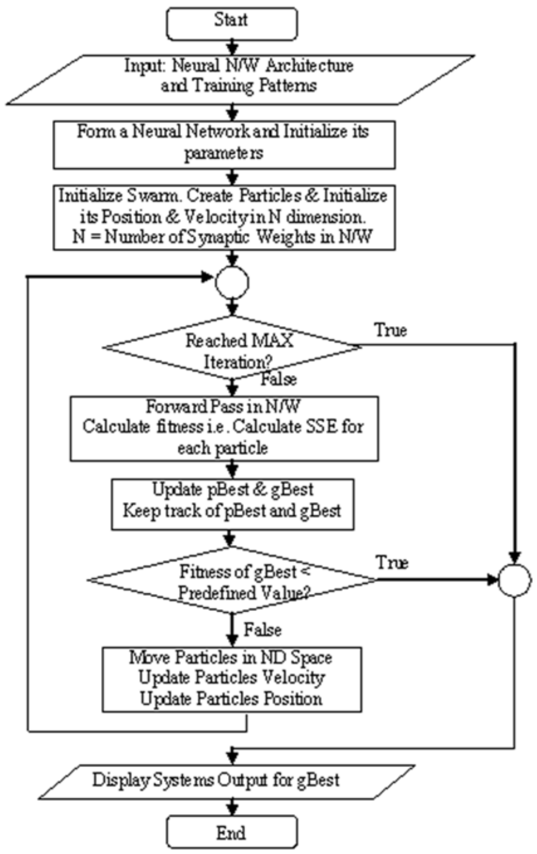
Start
Input: Neural N/W Architecture and Training Patterns
Form a Neural Network and Initialize its parameters
Initialize Swarm. Create Particles & Initialize its Position & Velocity in N dimension. N = Number of Synaptic Weights in N/W
Reached MAX Iteration?
True
False
Forward Pass in N/W Calculate fitness i.e. Calculate SSE for each particle
Update pBest & gBest Keep track of pBest and gBest
Fitness of gBest < Predefined Value?
True
False
Move Particles in ND Space Update Particles Velocity Update Particles Position
Display Systems Output for gBest
End

Figure 2. A Schematic of Neuro Swarm Optimization Algorithm

It is the flow diagram of neuro swarm optimization algorithm where the algorithm starts by taking network architecture and training pattern as input. Then the subsequent rectangle blocks represent the initialization of neural network synaptic weights and initialization of the particle of swarm. The diamond shape is decision block to decide whether the algorithm has reached to max iteration. Rectangular blocks next to it represent computation of fitness of particles and adjustment pBest and gBest value. The second diamond block in the figure checks the quality of the fitness value of the gBest particle and decides to iterate or to present the final output.

3.2 Implementation of neuro swarm optimization algorithm

The PSO formulae define each particle as potential solution in an N-dimensional space where N is equal to the total number of synaptic weights in the neural network.

3.2.1 Representation of particle in neuro swarm optimization algorithm

A particle in swarm is having two parameters ‘Position’ and ‘Velocity’. The Position of a particle is the location of that particle in the N-dimension space and particle moves with certain Velocity. The particle’s location ‘p_loc[N]’ and the particle’s velocity ‘p_vel[N]’ are one dimensional array of size ‘N’ where, ‘N’ is the dimension of the search space based on the problem characteristics.

3.2.2 Synaptic weight encoding

Let us consider that we have swarm of size five and neural network given in Figure 3 whose synaptic weights have to be encoded into particle. So the implementation of the swarm population can be defined as follows. The swarm is a vector containing class ‘particle’. The class ‘particle’ has two members, location and velocity. Location and velocity have been defined earlier. From the neural network shown in Figure 3 it is clear that the dimension of the problems is 8 i.e. N = 8. Table 3 is representing the encoding mechanism of the synaptic weights of the neural network appearing in Figure 3. P_1 is representing the 1st particle in the swarm of size five. The values of particles are initialized randomly. It is same as initializing the synaptic weights of neural network. In Table 3, the element ${}^4W_5$ indicates the randomly chosen value from the search space range between – 0.5 and +0.5 of the 5th element of the 4th particle in the swarm. The 5th element of the particle signifies that it is the 5th synaptic weight (W_5) of the network.

Table 3. The synaptic weight encoding mechanism

	Vectors (Swarm)				
	P_1	**P_2**	**P_3**	**P_4**	**P_5**
Synaptic Weights of neural network as elements of particles location	$^1W_1$	$^2W_1$	$^3W_1$	$^4W_1$	$^5W_1$
	$^1W_2$	$^2W_2$	$^3W_2$	$^4W_2$	$^5W_2$
	$^1W_3$	$^2W_3$	$^3W_3$	$^4W_3$	$^5W_3$
	$^1W_4$	$^2W_4$	$^3W_4$	$^4W_4$	$^5W_4$
	$^1W_5$	$^2W_5$	$^3W_5$	$^4W_5$	$^5W_5$
	$^1W_6$	$^2W_6$	$^3W_6$	$^4W_6$	$^5W_6$
	$^1W_7$	$^2W_7$	$^3W_7$	$^4W_7$	$^5W_7$
	$^1W_8$	$^2W_8$	$^3W_8$	$^4W_8$	$^5W_8$

Note The table represents the synaptic weight encoding mechanism. The vector/swarm contains particle P_i where each particle contains eight synaptic weight values. The ^iW_j means jth synaptic weight of the network vide Figure 3 is encoded as jth coordinate location value of the ith particle which is in eight dimension space with eight being the total number of synaptic weights of network shown in Figure 3.

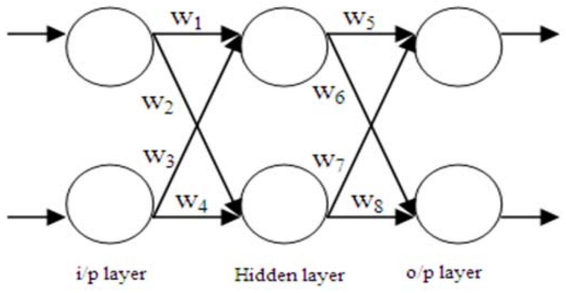

Figure 3. A Two Layer Feedforward Neural Network

This is a simple three layered feedforward neural network having the configuration 2 – 2 – 2 and it is having eight synaptic weights w_1 to w_8. This synaptic weight combination has to be encoded as particle location and decoded back to synaptic weight.

3.2.3 Updating particle velocity

While moving in the search space each particle remembers its best position reached so far represented as 'pBest' and they also know the group best maintained as 'gBest'. Each particle moves by updating their velocity considering their location of personal best location 'pBestL' and social best location 'gBestL'. Updating location of the particles signifies that the particle tries to reach an optimal solution. The pBest location of each particle can be stored in a vector 'pBestL'. Velocity of the ith particle is updated using Equation (4) [20]. The velocity of the ith particle is updated by the value of NewVel.

$$\text{NewVel}[k] = (W \times V[k]) + (R_1 \times C_1) \times (\text{pBestL}[k] - L[k]) + (R_2 \times C_2) \times (\text{gBestL}[k] - L[k]) \quad (4)$$

Where, k is the kth co-ordinate value into the N-dimension space; NewVel[k] = Updated Velocity of the ith particle in the kth direction; V[k] = Current Velocity of the ith particle in the kth direction; L[k] = Current value of the ith particle in kth dimension; W = Inertia factor; C_1 = Determine the relative influence of the cognitive component; C_2 = Determine the relative influence of the social component; pBestL[k] = pBest location value of the ith particle in kth direction; gBestL[k] = value location in the kth direction of gBest particle; R_1, R_2 = Random numbers, uniformly distributed over the interval [0, 1].

The random number R_1 and R_2 are used to preserve the assortment of the population. C_1 is a positive constant, called as coefficient of the self-recognition component; C_2 is a positive constant, called as coefficient of the social component [20]. The Equation (4), determines the next position of a particle, where, it is taking particles own experience into account, which is the particles best past position so far, and the experience of its most successful particle in the swarm. The inertia factor 'W' is controlling the effect of previous velocity on the new velocity. The value of the inertia factor 'W' is given by the Equation (5) [20].

$$W = W_{hi} - (t / Max_{iteration}) \times (W_{hi} - W_{lo}) \tag{5}$$

Where, W_{hi} = Initial weight; W_{lo} = Final weight; t = Current iteration number; $Max_{iteration}$ = Maximum iteration number.

3.2.4 Updating particle position

In the algorithm above Equation (5) is used to control the diversity of population and the diversification characteristic is gradually decreased. At a certain velocity, which gradually moves the current searching point close to pBest and gBest can be calculated [20]. The current position (searching point in the solution space) can be modified by means of the Equation (6). The location of the i^{th} particle updated using the Equation (6) [20]. The location of the i^{th} particle is updated by the value of NewPos.

$$NewPos[k] = l[k] + NewVel[k] \tag{6}$$

All swarm particles tend to move towards better positions. Hence, the best position i.e. optimum solution is expected to be obtained through the combined effort of the whole population.

3.2.5 Fitness computation of particles

The particle position values are decoded into the synaptic weights and for these synaptic weights the sum of squared error (SSE) induced by the neural network are computed. The sum of squared error becomes the fitness of the particle. On account of these fitness values the particles update their pBest and gBest subsequently. The formula of sum of squared error computation is given in expression (7) [24].

$$SSE = 1/2\sum\sum(O_{pi} - t_{pi})^2 \text{ for all p \&i} \tag{7}$$

Where, O_{pi} and t_{pi} are the actual and desired outputs respectively realized at the output layer, 'p' is the number of input pattern and 'i' is the number of nodes in the output layer.

3.2.6 Algorithm termination criteria

The algorithm terminates either when the sum of squared error reached to an acceptable minimum or when the algorithm completes its maximum allowed iteration.

4 Performance analysis

The neuro swarm optimization algorithm is implemented. The system is trained using the data sample mention in Table 2. Thereafter the performance of the neuro swarm optimization algorithm is analyzed. The performance of any algorithm is watched in terms of the quality of its result and the computational cost incurred to obtain that result. By doing the performance analysis we mean to observe the strength and weaknesses of neuro swarm optimization algorithm. We are presenting the performance analysis of the neuro swarm optimization algorithm with respect to its different parameter. In this section we also discuss the overhead of the neuro swarm optimization algorithm.

4.1 Performance analysis based on tuning different parameters

The parameters earmarked for the performance analysis of the neuro swarm algorithm are neural network configuration, population size (swarm size), number of generations (iterations), learning factor (C_1&C_2), and search space. We have analyzed the quality of the result/output, i.e. observing the average of sum of squared error taken for ten instances of SSE by the neuro swarm optimization algorithm. There is distinct performance analysis for different parameters.

4.1.1 Average sum of squared error vs. neural network configuration

Initially we want to find out optimum neural network configuration. So we need to observe the quality of the result obtained against the different neural network configurations. Keeping other parameters fixed at certain values, we change the number of nodes at hidden layer in the neural network to draw the performance characteristic of the algorithm. We set swarm size at hundred, number of generation at thousand, learning factors at 0.5 and initial search space range at [–0.5 to 0.5]. The Figure 4 exhibits the performance of neuro swarm optimization algorithm based on network configuration, where the x-axis indicates the change in the number of nodes at hidden layer (H) in (5 – H – 5) architecture and the y-axis indicates the average value of the sum of squared error obtained against different network configuration. It is already mentioned that the data set prepared using the sample mixture which is containing five gases, so we need to use five nodes at input layer and five nodes at output layer. Analysis based on network configuration is kind of structural training of the neural network. From Figure 4 it clearly derived that algorithm performs better for the network configuration 5 – 3 – 5 where value 3 indicates the number of nodes at hidden layer and for the entire configuration higher than 5 – 3 – 5 the performance of the algorithm is getting poorer and poorer.

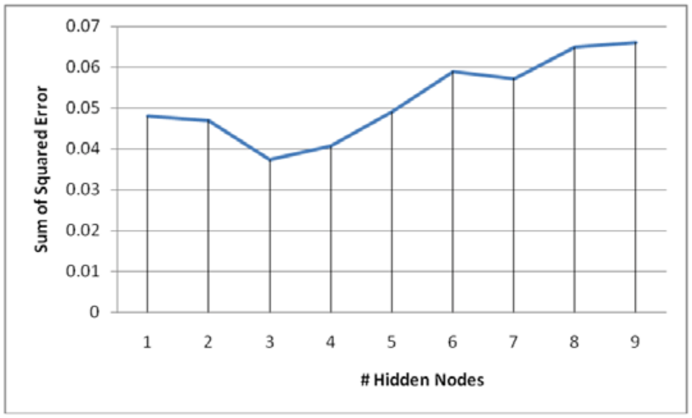

Figure 4. Average SSE vs. Neural Network Configuration

Along x-axis total number of nodes at hidden layer of the neural network is plotted and along y-axis average of ten instances of the sum of squared error is plotted. Among all the configurations, the configuration 5 – 3 – 5 performs best.

4.1.2 Average sum of squared error vs. population size (Swarm Size)

After the structural training of the network we need to establish the population size/ swarm size for the algorithm. Keeping other parameters fixed, we tune the swarm size to draw the performance characteristic of the algorithm based on swarm size. We set network configuration at 5 – 3 – 5, number of generations at thousand, learning factors at 0.5 and set initial search space range between –0.5 and 0.5. The Figure 5 exhibits the performance of neuro swarm optimization algorithm based on swarm size, where the x-axis indicates the change in the swarm size and the y-axis indicates the average value of the sum of squared error obtained against the different size of population taken. From the Figure 5 and 6 it is clearly visible that for the increasing values of the swarm sizes the performance of algorithm improves. At the population size equal to hundred the error gets down to below 5% and even less. So we set the population size at hundred particles for analyzing performance of algorithm for other parameters.

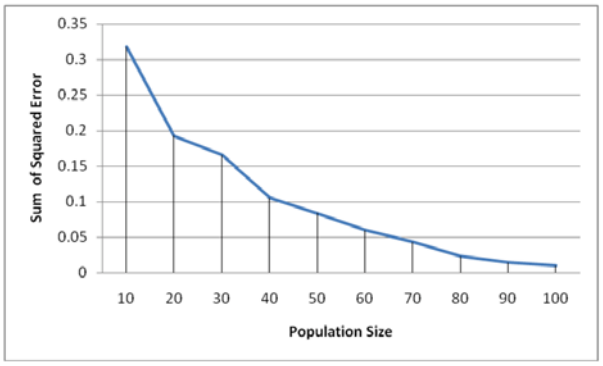

Figure 5. Average SSE vs. Population Size

Along x-axis swarm size is plotted and along y-axis average of ten instances of the sum of squared error is plotted. The value of average sum of squared error is falling down monotonically . For swarm size equal to hundred errors reduces below 5%.

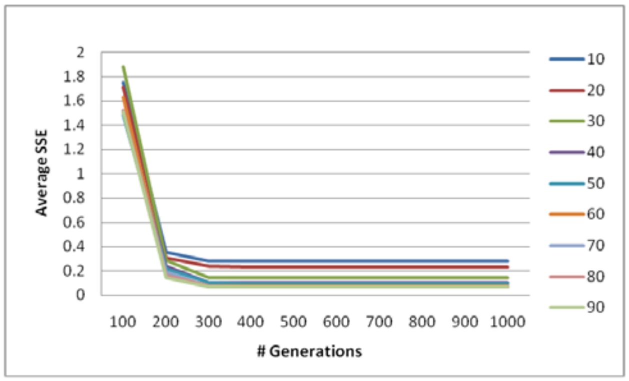

Figure 6. Average SSE vs. Number of Generations for different Population Size

Along x-axis swarm size is plotted and along y-axis average of ten instances of the sum of squared error is plotted. The figure is showing that on increasing swarm size the performance of neuro swarm algorithm improves. The darkest line indicated lowest swarm size and brightest line indicates highest value of swarm size.

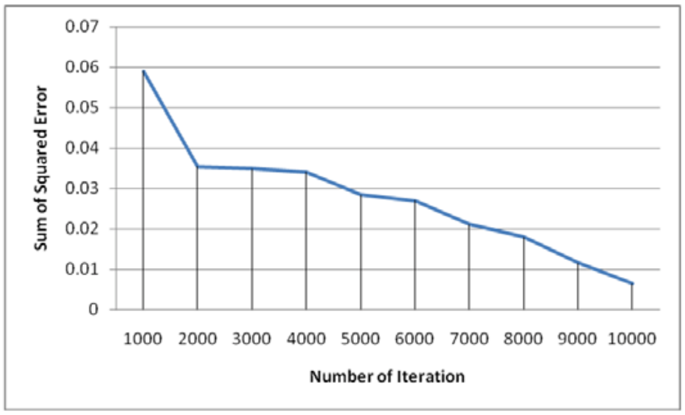

Figure 7. Average SSE vs. Number of Generations

Along x-axis generation number is plotted and along y-axis average of ten instances of the sum of squared error is plotted. When number of generation increases, the value of average SSE decreases. Initially at thousandth iteration average SSE is about 6% and it is falling down on increasing the value of total number of generation finally it fall down below 1% at ten thousandth iteration

4.1.3 Average sum of squared error vs. number of generations

The Figure 7 exhibits the performance of particle swarm optimization algorithm against different number of generations. Here the average sum of squared error is computed against various iterations. We set the population size at hundred, the network configuration at 5 – 3 – 5, learning factors to 0.5 and the search space range at [–0.5 to 0.5]. From the graph vide Figure 7 we found that the sum of squared error decreases for all the increasing values of generations.

4.1.4 Average sum of squared error vs. learning factor

C_1 and C_2 are positive constant, called as coefficient of the self-recognition component and coefficient of the social component respectively. These are also known as learning factor, which control the velocity of the particle in the swarm optimization algorithm. Keeping other parameters fixed, we vary $C_1 = C_2$ in between 0.3 and 1.9 to observe the performance characteristic of the algorithm based on the learning factor. We set network configuration to 5 – 3 – 5, population size at hundred, number of generation at thousand, and initial search space range between –0.5 and 0.5. Figure 8 exhibits the performance of neuro swarm optimization algorithm based on learning factors, where x-axis indicates the change in the value of learning factors and y-axis indicates the average value of the sum of squared error obtained against different value of learning factor. It is found that initially the performance of algorithm improves for the increasing values of learning factor but increasing its values after C_1=C_2 = 1.1 the performance of the algorithm has started falling.

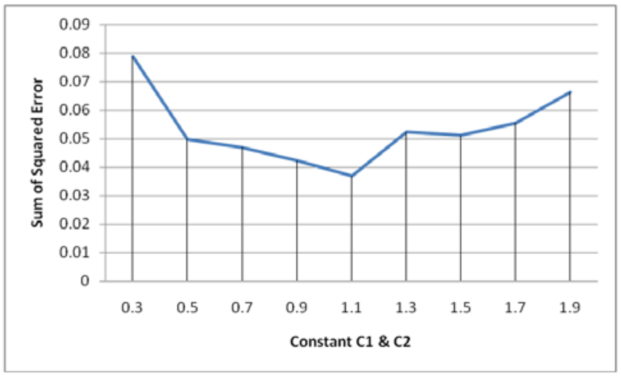

Figure 8. Average SSE vs. Learning Factor

Along x-axis learning factor is plotted and along y-axis average of ten instances of the sum of squared error is plotted. Here it is observed that average SSE is lowest for $C_1 = C_2 = 1.1$ and it is increases for both lower and higher value of learning factor.

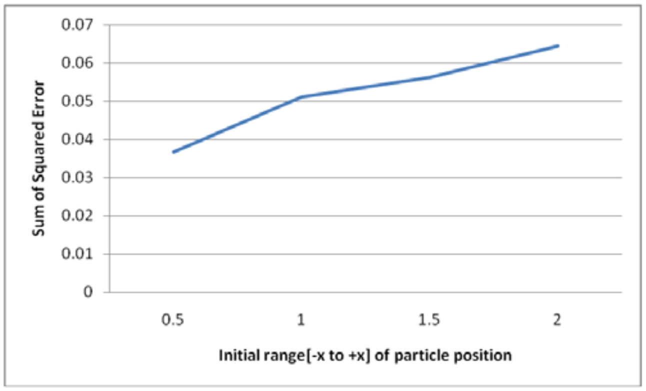

Figure 9. Average SSE vs. Initial Solution Space

Along x-axis range of search space is plotted and along y-axis average of ten instances of the sum of squared error is plotted. Here value 0.5 along x-axis represents range of the search space and it is between [– 0.5 and +0.5]. It is observed that average SSE increases linearly when the range of initial search space is widening.

4.1.5 Average sum of squared error vs. search space

After analyzing the performance of the algorithm for the parameters like population size, network configuration, learning factors it becomes necessary to check the performance of the algorithm against the search space, because to some extant the performance of the neural network depends on the initial choice of synaptic weights. Figure 9 exhibits the performance of neuro swarm optimization algorithm based on initial choice solution space range, where x-axis indicates the change in the choice of range for initial solution and y-axis indicates the average value of the sum of squared error obtained against different range of initial solution. Analyzing the graph in Figure 9 it is derived that the performance of algorithm is coming down on widening the search space meaning that the average sum of squared error continues to increases each time the range widens.

4.2 Comparing PSO based NN training and other standard method

The PSO based neural network training method has its own advantages and disadvantages and the standard neural network training methods such as backpropagation algorithm has its own advantages and disadvantages. Contrasting those advantages and disadvantages between them we can establish the superiority of any one of them. The backpropagation algorithm has many variants like steepest decent method, conjugate gradient method etc. In steepest decent method the algorithm adjusts the weights in the steepest descent direction (negative of the gradient) and in the conjugate gradient algorithm a search is performed along conjugate directions. It is known that the conjugate gradient method has some plus point over steepest decent method so we are choosing conjugate gradient method for comparison with our PSO based neural network training with requite parameter settings for both the algorithms. In Figure 10 we have plotted numbers of generations along x-axis and average sum of squared error is plotted along y-axis. From Figure 10 it is found that initially the performance of conjugate gradient based neural network training performs poorer than PSO based neural network training. It is also observed that the convergence speed is faster in PSO based training method. At hundredth iteration average SSE is about 12% for PSO based NN training algorithm and 17% for conjugate gradient based NN training algorithm. From the figure we observed that the performance quality gap between both the algorithms are reduces for increasing values of iterations. At thousandth iteration both algorithms tend to produce almost same result and the performance quality gap between the algorithms reduces to zero. The average SSE reduces to value less than 1%. This is because contrary the PSO is a population based algorithm where there is greater chance of selecting fittest individual among the whole population the conjugate gradient bases NN training starts with single guess of weight vector. The PSO has more parameters to adjust or to tune in comparison to conjugate gradient method. So again it opens an area where we can regulate the performance of the algorithm by tuning the parameters at some optimum values.

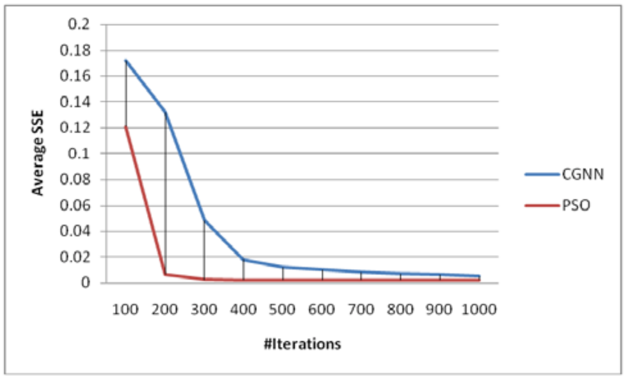

Figure 10. Average SSE vs. Iteration

Along x-axis generation number is plotted and along y-axis average sum of squared error is plotted. For the increasing number of generations the value of average SSE decreases. The blue line represents the curve for CGNN (conjugate gradient based neural network) algorithm. The first reading taken at hundredth iteration and maximum iteration considered in comparison is thousand. Initially at hundredth iteration thousandth iteration average SSE is about 12% for PSO based NN training algorithm and 17% for CGNN based NN training algorithm. At thousandth iteration both algorithm tends to produce almost same result. And for both the algorithm error reduces to very-very less than 1%.

4.3 Computational complexity

The computation at complexity of the neuro swarm algorithm is same as the computational complexity of particle swarm algorithm. In [4, 26] the corresponding authors briefly discuss about the computational complexity of the particle swarm optimization algorithm. The computational complexity of the particle swarm algorithm is due to fitness function we used. So the computational complexity of neuro swarm algorithm entirely depends on the number of times fitness function is evaluated and the total cost to evaluate a fitness function in single instance. Let us assume that fitness function evaluated 'f' number of time and 'k' is the cost needed to evaluate fitness function. So the computational overhead is given as $f \times k$. In particle swarm optimization algorithm the fitness function is evaluated $m \times (g + 1)$ numbers of time where 'm' is the total number of particle in swarm and 'g' is the total number of generations/iterations. The fitness function considered in this case is nothing but the computation of the sum of squared error induced by neural network for given training pattern. From expression (7) it is clear that to compute sum of squared error we have to compute square error of the error obtained at output layer for each pattern 'p'. The computation of error at output layer is nothing but two dimensional matrix multiplications. The cost of to computing two dimensional matrix multiplications is $O(n^2)$. So the cost of computing sum of squared error is $k = p \times O(n^2)$ where 'p' is the total number of patterns. Hence the total computational overhead of neuro swarm algorithm stands to $m \times (g+1) \times k$.

5 Results

From the prepared data samples we consider 80% samples for training and 20% for testing purpose. The output of the network is denormalized according to Equation (3) to report systems output in terms of the concentration of the gas component present in the gaseous mixture. Here we are providing a sample test result obtained for the input sample 2 which is given in Table 2. There, the input vector is [0.260, 0.346, 0.240, 0.142, 0.843] and target vector is [0.01, 0.02, 0.02, 0.02, 1.0]. Finally we are producing this input vector for giving input to the neural network for testing purpose. The test result of the neural network for this input vector is shown in Table 3.

Table 4. System result presentation

Testing i/por i/p node no.	Normalized i/p value(Sample 2 of Table 2)	n/w Actual o/p	System's o/p in PPM
1	0.260	0.016	0080
2	0.346	0.022	0110
3	0.240	0.023	0115
4	0.142	0.022	0110
5	0.843	0.993	4965

Note. The first column represents the node number at input layer. The values shown in second column are normalized value of sensors response of gas mixture sample 2 of Table 2.The third column represents the output which is directly obtained from the neural network and its denormalized value is shown in the fourth column.

6 Conclusion

The article presents the mechanisms for the development of an intelligent sensory system comprising semiconductor based gas sensor array and neural network classifier. Together they offer solution to the detection of gases in manhole gas mixture. The gas detection problem is treatded as a pattern recognition problem. We have shown how a neural network is trained using particle swarm optimization algorithm. The article provides detail study on the implementation overview of the neuro swarm optimization algorithm. The article also addresses the significance of the cross-sensitivity issue. The cross-sensitivity effect from the data sample is filtered out by the training of the neural network classifier. The neural network has been trained such that it can forecast actual concentration level of the gas components present in manhole gas mixture. In the performance analysis section we have provided the pros and cons of the neuro swarm algorithm. The performance analysis is based on the tuning of different parameters of the algorithm. The article devoted study on the evaluation of computational expensiveness of neuro swarm algorithm. Finally we provide clear idea to present the systems output in terms of the concentration of gases present in manhole gas mixture. Comparing the obtained system output and the safety limits of the corresponding gases system may predict alarming situation.

References

[1] Barsky J. B. Simultaneous Multi-Instrumental Monitoring of Vapors in Sewer Headspaces by Several Direct-Reading Instrument, Environmental Research. 1986; 39(2): 307-320. http://dx.doi.org/10.1016/S0013-9351(86)80057-3

[2] Chatchawal, A & Wisitsoraatb, Portable electronic nose based on carbon nanotube-SnO2 gas sensors and its application for detection of methanol contamination in whiskeys, Sensors and Actuators B: Chemical, SNB-12243, 2010.

[3] Eduard Llobet, &RaduIonescu, Multi-component Gas Mixture Analysis Using a Single Tin Oxide Sensor and Dynamic Pattern Recognition, IEEE SENSORS JOURNAL. 2001; 1(3). http://dx.doi.org/10.1109/JSEN.2001.954833

[4] Escalante, H. J. Montes, M. & Sucar, L. E., Particle Swarm Model Selection, Journal of Machine Learning Research. 2009; 405-440.

[5] Gary M. H., Reference Data Sheet on Gas(es), Meridian Engineering and Technology. Available from: http://www.meridianeng.com/sewergas.html, 1993.

[6] GeorgiosTsirigotis, Neural Network Based Recognition, of CO and NH3 Reducing Gases, Using a Metallic Oxide Gas Sensor Array, Scientific Proceedings of RTU. Series 7. Telecommunications and Electronics. 2003; 3.

[7] Jing-Ru Zhang, Jun Zhang, Tat-Ming Lok, Michael R. Lyu, A hybrid particle swarm optimization–back-propagation algorithm for feedforward neural network training, Applied Mathematics and Computation. 2007; 185(2): 1026-37. http://dx.doi.org/10.1016/j.amc.2006.07.025

[8] Jun Li, A Mixed Gas Sensor System Based on Thin Film SAW Sensor Array and Neural Network, 7803-0976-6/93/3: l, 1993IEEE.

[9] Kennedy, J. & Blackwell, T. Particle swarm optimization An Overview, Springer Science + Business Media, LLC, 2007.

[10] Kennedy, J. & Russell C. E., Swarm Intelligence, Morgan Kaufmann Publishers, ISBN 1-55860-595-9, 2001.

[11] Lewis, Dangerous Properties of Industrial Materials, (9th edition) Volumes 1-3. New York, Van Nostrand Reinhold, 1996.

[12] Maxime Ambard, A Spiking Neural Network for Gas Discrimination using a Tin Oxide Sensor Array, 4th IEEE International Symposium on Electronic Design, Test & Applications,0-7695-3110-5/08/IEEE, 2008.

[13] NIOSH, Volunteer Fire Fighter Dies during Attempted Rescue of Utility Worker from a Confined Space. Available from: http://www.cdc.gov/niosh/fire/reports/face201031.html, 2011.

[14] Ojha, V. K. Dutta, P.Saha, H. Ghosh, &Ghosh, S. Linear regression based statistical approach for detecting proportion of component gases in manhole gas mixture, International Symposium on Physics and Technology of Sensors 2012, in press. http://dx.doi.org/10.1109/ISPTS.2012.6260865

[15] Ojha, V. K. Dutta, P. &Saha, H. Detection of proportion of different gas components present in manhole gas mixture using backpropagation neural network, Intentional Conference on information & Network Technology. 2012; 37: 11-15.

[16] Ojha, V. K. Dutta, P. K. Saha H. & Ghosh, S. Application of Real Valued Neuro Genetic Algorithm in Detection of Components Present in Manhole Gas Mixture, Advances in Intelligent and Soft Computing. 2012; 166: 333-340. http://dx.doi.org/10.1007/978-3-642-30157-5_33

[17] Ojha, V. K.Dutta, P. Saha, H. S & Ghosh, S. A Neuro-Swarm Technique for the Detection of Proportion of Components in Manhole Gas Mixture, In proceedings of International Conference on Modeling, Optimization and Computing. 2012; 2: 1211-18.

[18] Ojha, V. K.Dutta, P. Saha, H. S & Ghosh, S. A Novel Neuro Simulated Annealing Algorithm for Detecting Proportion of Component Gases in Manhole Gas Mixture, International Conference on Advances in Computing and Communications. http://dx.doi.org/10.1109/ICACC.2012.54.

[19] Ojha, V. K.Dutta, P. Saha, H. Performance Analysis Of Neuro Genetic Algorithm Applied On Detecting Proportion Of Components In Manhole Gas Mixture, International Journal of Artificial Intelligence & Applications (IJAIA). 2012; 3(4). http://dx.doi.org/10.5121/ijaia.2012.3406.

[20] Premalatha, K. &Natarajan, A. M, Hybrid PSO and GA for Global Maximization, International Journal on Open Problems Compt. Math. 2009; 2(4): 1998-6262.

[21] R. Akbari, and K. Ziarati, A Rank Based Particle Swarm optimization with Dynamic Adaptation, Journal of Computational and Applied Mathematics, Elsevier. 2011; 235(8): 2694-2714. http://dx.doi.org/10.1016/j.cam.2010.11.021

[22] Rakesh Malviya, Dilip Kumar Pratihar, Tuning of neural networks using particle swarm optimization to model MIG welding process, Swarm and Evolutionary Computation. 2011; 1(4): 223-235. http://dx.doi.org/10.1016/j.swevo.2011.07.001

[23] Simon H. Neural Network a Comprehensive Foundation, (2nd edition), Pearson Prentice Hall, 2005.

[24] Sivanadam, S. N. &Deepa S. N. Principles of Soft Computing, (1st edition), Wiley India (p) Ltd. 2007.

[25] Wu Pan, Ning Li, &Pandeng Liu, Application of Electronic Nose in Gas Mixture Quantitative Detection, Proceedings of IC-NIDC-2009, 978-1-4244-4900-2/09/IEEE, 2009.

[26] Yakubu S. Baguda, Low Complexity PSO-Based Multi-objective Algorithm for Delay-Constraint Applications, ICIEIS 2011, Part III, CCIS 253. 2011; 274-283.