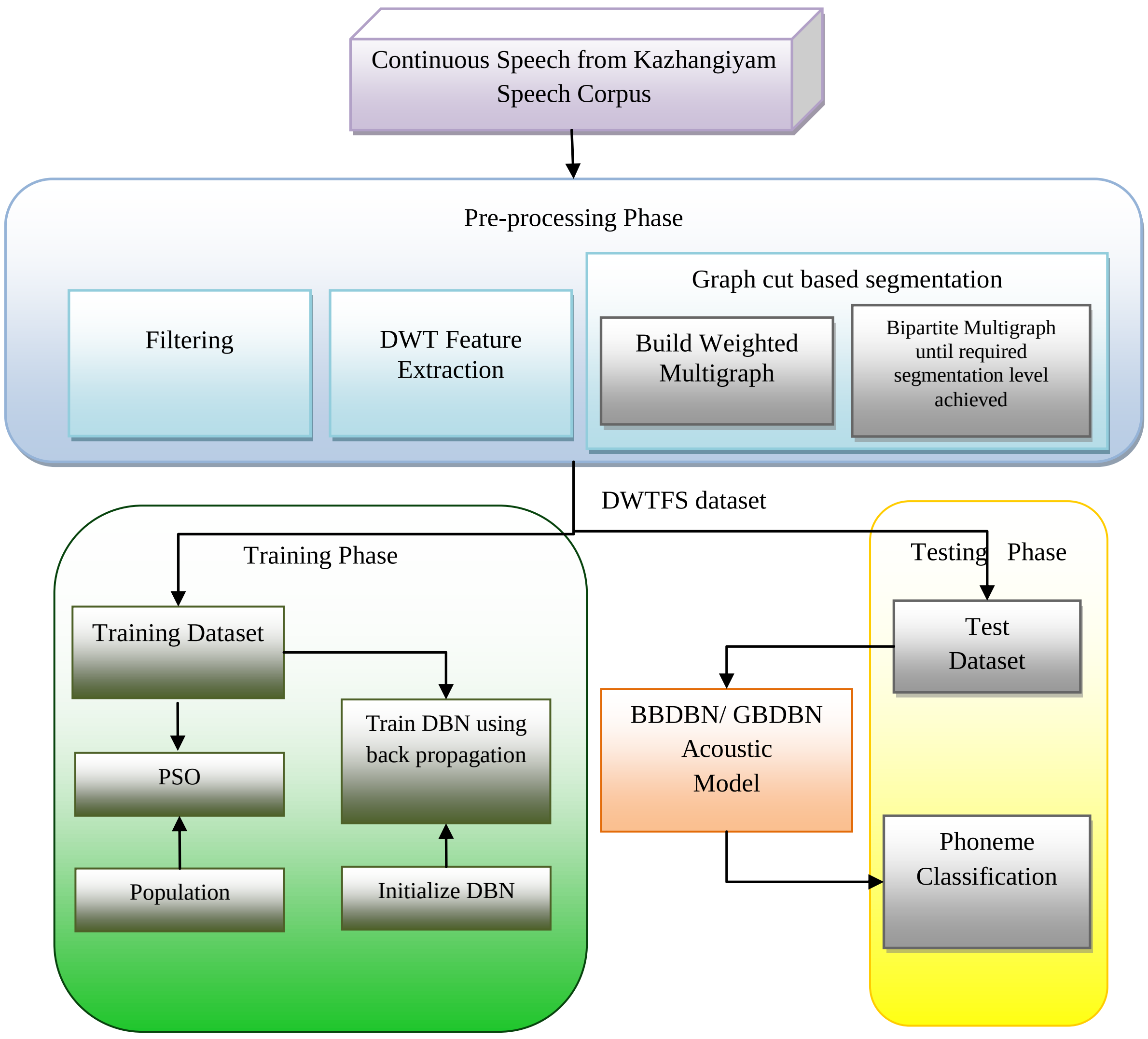
6. PSO PRE-TRAINED DBN FOR PHONEME RECOGNITION

Learning in neural network vastly pivot on the parameter initialization of the network. Parameters of the network generally are assigned with random initialization. Performing pretraining provides better initialization to neural networks and helps in developing more optimal model. The time complexity involved in pre-training and training the DBN increases with the number of layers in the DBN. It is obvious that if the RBMs in the DBN are trained in isolation using contrastive divergence as discussed in the previous chapter, the time complexity to train the DBN increases with the increased depth. Further in traditional approach, random initialization of network parameters and usage of gradient descent technique for learning may suffer with the solution being trapped in local minima. This chapter discusses the proposed architecture to build Particle Swarm Optimization pretrained Deep Belief Networks (PSO-DBN) model for Tamil phoneme recognition. This uses PSO to pre-train all the layers of DBN in a single pass and helps to reduce the time complexity theoretically and enable the movement of particles towards global minima rather than being trapped in local minima. This chapter also discusses on the performance of DBN phoneme recognition model pretrained using PSO variants namely, Second Generation PSO (SGPSO) and New Method PSO (NMPSO) in addition to the basic PSO.

6.1 TAMIL PHONEME RECOGNITION MODEL USING PSO-DBN

The overall framework of the proposed PSO-DBN based acoustic model is shown in Fig.6.1. The framework comprises of three phases: pre-processing phase, training phase and testing phase. Continuous speech samples collected is directly fed into the pre-processing phase where the speech wave form is segmented into phonetic segments using graph cut based segmentation algorithm as discussed in chapter 3. The DWTFS dataset built using graph cut based segmentation is fed to the training phase to build various PSO based DBN acoustic models. The acoustic models are then tested using the test dataset and the model efficiencies are studied through RMSE, PER, precision, recall, F-measure and accuracy.



Continuous Speech from Kazhangiyam Speech Corpus
Pre-processing Phase
Filtering
DWT Feature Extraction
Graph cut based segmentation
Build Weighted Multigraph
Bipartite Multigraph until required segmentation level achieved
DWTFS dataset
Training Phase
Training Dataset
PSO
Population
Train DBN using back propagation
Initialize DBN
BBDBN/ GBDBN Acoustic Model
Testing Phase
Test Dataset
Phoneme Classification

Fig 6.1 Architecture of PSO-DBN Acoustic Model

The training phase in building PSO-DBN involves pretraining with PSO followed by back propagation based fine tuning subphases. The pretraining algorithm used to optimize the parameters of DBN using PSO is given below. The neurons in the DBN are defined with the sigmoid activation function. The algorithm takes the hyper parameters of DBN number of layers, number of neurons in each layer and the training dataset as input and returns the evolved population set and the best individual in the population as output.

Algorithm 6.1 Optimizing DBN parameters using PSO

Step 1: Initialize the PSO parameters namely, inertia coefficient (ω), personal acceleration coefficient (c_1) and global acceleration coefficient (c_2)

Step 2: Instantiate population P of size M whose individuals are of the form as shown in

Fig. 6.2

Step 3: for all individuals i in the population P do

a. Initialize the position vector *($x_i(1)$)* as uniform random values

b. Initialize the velocity vector *($v_i(1)$)* as zero

Step 4: for each generation *t* do

Step 5: for each individual *i* in the population *P* do

a. Calculate the new velocity,

$$v_i(t+1)=wv_i(t)+h(t)v_i(t)+c_1r_1(p_i(t)-x_i(t))+c_2r_2(p_g(t)-x_i(t)$$

b. Calculate the new position, $x_i(t+1) = x_i(t) + v_i(t+1)$

c. Evaluate the cost of the individual for the given training dataset using algorithm 6.2

d. Update the personal best cost and position (pi)

e. Update the global best cost and position (pg)

f. Update inertia (ω)

W_1	B_1	W_2	B_2	...	W_{n-1}	B_{n-1}

Fig. 6.2 Coding scheme of each individual of the population

Various parameters that model the characteristics of the DBN are received as input by the pretraining algorithm. Synaptic weights w_{ij} of the edges connecting the neurons of successive layers, bias values b_j of each neuron, the activation function *f(x)* defining the neurons are such parameters. The connection weights and bias parameters of the DBN are codified as an individual in the population of PSO to represent the solution of problem under consideration.

The codification of a DBN in a PSO is done as follows. The population *P* of size *M* is represented as a collection of individuals each representing the parameters of a DBN model. The coding scheme used to represent each individual is shown in Fig. 6.2 which is of length

$\sum_{i=1}^{L-1} N_{i+1}(N_i+1)$, where $L-1$ denotes the number of RBMs in the DBN with L defining the number of DBN layers, N_i and N_{i+1}, the number of neurons in input and output layers of i^{th} RBM. The vectors W_i and B_i in the coding scheme denotes the connection weights and biases of output neurons of i^{th} RBM, R_i, where $1 \leq i < L$. Codification of the solution to the problem for representing the network parameters as required by PSO is defined in equation 6.1. Each row of the matrix represents an individual in the population.

$$\begin{bmatrix} x_{1,1} & \cdots & x_{1,n_1\times n_2} & x_{1,n_1\times n_2+1} & \cdots & x_{1,q} \\ \vdots & \ddots & \vdots & \vdots & & \vdots \\ x_{S,1} & \cdots & x_{S,n_1\times n_2} & x_{S,n_1\times n_2+1} & \cdots & x_{S,q} \end{bmatrix}, q = \sum_{i=1}^{L-1} n_i \times n_{i+1} + n_i \qquad (6.1)$$

$\forall$ R_i, i and $i+1$ being the input and output layers, $n_i \times n_{i+1}$ elements of weight matrix and n_{i+1}, the number of biases for neurons in layer $i+1$, are located from $x_{k,\sum_{j=1}^{i-1} n_j \times n_{j+1} + n_j + 1}$ to $x_{k,\sum_{j=1}^{i} n_j \times n_{j+1} + n_j}$ in equation 6.1 for any individual k, where $1 \leq k \leq M$.

Each individual in the population maintains its position vector, velocity vector and local best position. The codification scheme portrayed in Fig. 6.2 defines the position vector and velocity vector of each individual in PSO population which in turn represents the DBN parameters. The position vector, velocity vector and the local best position of each individual is initialized with random values. Then each individual in the population is updated for t generations. In each generation the velocity of the particle is updated as given in step 5(a) of the algorithm, followed by the updation of the current position. At the end of each generation, the cost of each individual in the current population is evaluated using algorithm 6.2. Then the local best of each individual and global best of the population are updated.

Algorithm 6.2 Evaluating the cost of each individuals

Step 1: for each individual i in population do

Step 2: Construct a DBN *(₦ᵢ)* by transforming individual i coded as in Fig. 6.2 into a DBN structure

Step 3: Pass the training dataset through the layers of ₦ᵢ using $p(h_j = 1|v,\theta) = \sigma(a_j + \sum_{i=1}^{V} w_{ij} v_i)$

Step 4: Evaluate $Cost(₦_i) =(y_i - o_i)^2/m$, where y_i, o_i and m are predicted output vector, actual output vector and number of training samples.

Step 5: end for

The algorithm 6.2 evaluates the individuals in the population that codifies the respective candidate DBNs. The algorithm receives the current population of PSO as input and returns the cost of DBNs as output. The performance of deep neural networks is decided based on their fitness values. The algorithm initially transforms individuals to DBN and then evaluates its cost by passing the training set to DBN. Mean Square Error (MSE) is used as the fitness or cost function and the problem is defined as a minimization problem on the cost function defined by MSE. The cost function is given as follows:

$$CF = \left(\frac{1}{p.C}\right)\sum_{\Phi=1}^{p}\sum_{i=1}^{C}\left(o_i^{\Phi} - y_i^{\Phi}\right)^2 \tag{6.2}$$

where p is the dataset size, C is the number of output classes, o_i^{ϕ} denotes the desired output, and y_i^{ϕ} denotes the acquired output. The PSO pretraining phase is succeeded with the back propagation as explained in section 5.1 and the PSO-DBN acoustic model is developed.

The basic PSO is extended with its variations namely, SGPSO and NMPSO to search and identify better solutions for any problem considered. In this work, these two PSO variants are also used for pretraining DBN and to build the SGPSO-DBN and NMPSO-DBN acoustic models.

Second Generation Particle Swarm Optimization (SGPSO)

SGPSO is an improvement to the basic PSO. The basic PSO works on maintaining the local optimum of each particle and global best solution to help the population migrate towards the optimum solution. In SGPSO, a third parameter depicting the geometric centre of the optimum swarm in addition to the other two the local best and global best is included to identify the optimum solution. The inspiration behind the idea is that, the birds maintain a certain distance between its swarm centre from the location of the food. The bird flock stays around a particular area for a period of time with a fixed swarm centre for each member bird. Once that area is explored, it moves to the new area for food search, in turn fixing its new geometric centre. The geometric centre $\bar{P} \in \mathbb{R}^D$ of the swarm is updated once for every T time, as follows:

$$\bar{P} = \frac{1}{M}\sum_{i=1}^{M} p_i \tag{6.3}$$

where M is the number of particles in the population. The velocity calculation in SGPSO is given below,

$$v_i(t+1) = \omega v_i(t) + c_1 r_1 \big(p_i(t) - x_i(t)\big) + c_2 r_2 \left(p_g(t) - x_i(t)\right) + c_3 r_3 \big(\bar{P} - x_i(t)\big) \qquad (6.4)$$

where c_1, c_2 and c_3 are local, global and geometric centre acceleration coefficients, r_1, r_2 and r_3 are random number in the range [0, 1] and ω is the velocity inertia. Thus the algorithm 6.1 is updated for SGPSO and is given below. The SGPSO algorithm also takes number of DBN layers, number of neurons in each layer and the dataset to train as input and the evolved population set and the best individual in the population as output.

Algorithm 6.3 Optimizing DBN parameters using SGPSO

```
Step 1: Initialize the PSO parameters namely, inertia coefficient $\omega$, personal acceleration
        coefficient $c_1$, global acceleration coefficient $c_2$ and geometric centre $c_3$
Step 2: Instantiate population P of size M whose individuals are of the form as shown in Fig.
        6.2
Step 3: for all individuals i in the population P do
        a. Initialize the position vector $x_i(1)$ as uniform random values
        b. Initialize the velocity vector $v_i(1)$ as zero
Step 4: for each generation t do
Step 5: for each individual i in the population P do
        a. Calculate the new velocity,
              $v_i(t+1) = \omega v_i(t) + c_1 r_1 \big(p_i(t) - x_i(t)\big) + c_2 r_2 \left(p_g(t) - x_i(t)\right) + c_3 r_3 \big(\bar{P} - x_i(t)\big)$
        b. Calculate the new position, $x_i(t+1) = x_i(t) + v_i(t+1)$
        c. Evaluate the cost of the individual for the given training dataset using algorithm 6.2
        d. Update the personal best cost and position $p_i$
        e. Update the global best cost and position $p_g$
        f. Update inertia $\omega$
```

In the above algorithm, step 5(a) shows the velocity update done in generation of SGPSO. The fourth term in the velocity update equation $c_3 r_3(\overline{P} - x_i(t))$ defines the geometric term that incorporates the influence of geometric certre of the population in velocity evolution. In this term c_3 and $r_3$denotes the coefficient of geometric centre and a random value respectively.

New Model Particle Swarm Optimization (NMPSO)

The algorithm NMPSO is an integrated approach of various PSO. The idea of linearly varying inertia weight on a course of generation proposed in [86] is used to control the inertia. It significantly improves the fit of model parameters over the basic PSO algorithm. The inertia weight is linearly varied using the following equation,

$$\omega = (\omega_1 - \omega_2) \times \frac{\mathrm{MAXITER-iter}}{\mathrm{MAXITER}} + \omega_2 \tag{6.5}$$

where ω_1 and ω_2 are the initial and final inertia weights respectively, MAXITER is the maximum number of iterations and *iter* is the current iteration number.

The concept of velocity resting developed in [87] is used when there is no improvement in global best position with the increasing generations. The velocity resting is calculated as follows,

$$v_i = v_i + (2 \times r - 1) \times v_{max} \tag{6.6}$$

where *r* is the uniformly distributed random number in the range [0,1] and v_{max} is the maximum random perturbation magnitude to each selected particle dimension.

Crossover operator proposed in [88] is defined as follows,

$$ch_1(x_i) = r_i par_1(x_i) + (1 - r_i) par_2(x_i) \tag{6.7}$$

which is used in NMPSO to perform crossover at a rate α. Here, r_i is a uniformly distributed random number in the range [0, 1], ch_1 is the offspring and par_1 and par_2, the parents of the offspring. The velocity of offspring is calculated as follows,

$$ch_1(v_i) = \frac{par_1(v_i)+par_2(v_i)}{|par_1(v_i)+par_2(v_i)|} |par_1(v_i)| \tag{6.8}$$

The algorithm also uses a gaussian mutation operator proposed in [89] at a mutation rate β. The parent *par* to create an offspring is chosen randomly. The mutation is performed using the following equation,

$$ch(x_i) = par(x_i) + \frac{MAXITER-iter}{MAXITER} N(0,1) \tag{6.9}$$

for N being a Gaussian distribution, *MAXITER* the maximum number of allowable iterations and *iter,* the current iteration.

The concept of dynamic random neighborhood used by [90] is applied in NMPSO at a rate of γ. This involves dividing the population into neighborhoods, with each neighborhood having a maximum population of MAXNEIGH. The best member of each neighborhood $p_{g_{K_n}}$ is computed. Then the velocity of each particle is updated in terms of neighborhoods as follows:

$$v_i(t+1) = \omega v_i(t) + c_1 r_1 \big(p_i(t) - x_i(t)\big) + c_2 r_2 \left(p_{g_{K_n}}(t) - x_i(t)\right) \tag{6.10}$$

∀ neighborhoods. So, the NMPSO is a combination of several improvements on PSO that includes varying inertia, crossover offspring, particle mutation and dynamic neighborhoods. The flow of NMPSO is listed out in the following algorithm which too takes training dataset and DBN hyperparameters as input and presents the global best individual as a result.

Algorithm 6.4 NMPSO for optimizing DBN parameters

```
Step 1: Initialize the population given as a set of individuals, $x_i$, where $i=1, 2, ....M$.
Step 2: Do until the stop criteria is reached
Step 3: If $p_g$, does not improve for a said number of iterations then apply velocity resting as
        defined in equation 6.6
          If random(0,1)>γ the create new neighborhoods
Step 4: Update inertia weight using equation 6.5
Step 5: For all individual $x_i$, do
          Calculate their fitness
Step 6: For all individual $x_i$, do
          Update its best position $p_i$
Step 7: For each neighborhood $K_j$, do
          Update the best individual $p_{g_{K_n}}$
Step 8: For all dimensions, for all individuals do
      a. Update velocity using equation 6.10
      b. Update current position $\mathrm{x_i(t+1)}$
      c. If random(0,1)>α, perform crossover using equation 6.7 and 6.8
      d. If random(0,1)>β, perform mutation using equation 6.9
```

The output of any PSO pretraining provide the global best solution in the problem space, which is further decoded to initialize the weights of the links connecting the layers of DBN and biases of neurons in each layer of DBN to form PSO-DBN, SGPSO-DBN and NMPSO-DBN models respectively. The DBN models are then trained with the back propagation training to build efficient acoustic models.

6.2 EXPERIMENT AND RESULTS

As in previous experiments the DWTFS dataset is used to implement phoneme recognition models PSO-DBN, SGPSO-DBN and NMPSO-DBN. Experiments are performed by varying the number of layers and the population involved in identifying the optimal solution for the phoneme recognition problem. The default values of parameters common to all three PSOs were set as follows: a population of 25, inertia co-efficient 1, damping inertia co-efficient 0.99, personal acceleration and social acceleration co-efficients 2, and a maximum of 500 generations. For SGPSO the geometric centre acceleration co-efficient has been set to 2, while the social and personal accelerations co-efficients were assigned 2.05, and the geometric centre updating time as 20. For NMPSO pre-training, the parameters were defined as follows: the population size 100, number of neighbourhoods 4, initial and final inertia coefficients 0.9 and 0.4, crossover and mutation factors 0.1, and neighborhood updation rate 0.2. The number of neurons in the 7-layer DBN is set as 90x100x120x120x100x70x39. The sigmoid activation function is used for firing the neurons.

The best cost observed over the generations of PSO pretraining of a 7-layer DBN when trained with population sizes 5, 10, 15, 20, 25, 30 and 35 is shown in Fig. 6.3. The best cost observed shows that the learning becomes better while increasing the population size but started degrading after the population size of 25. The learning observed in the curve of PSO pretraining with population 25 shows a good improved in the best cost achieved over generations.

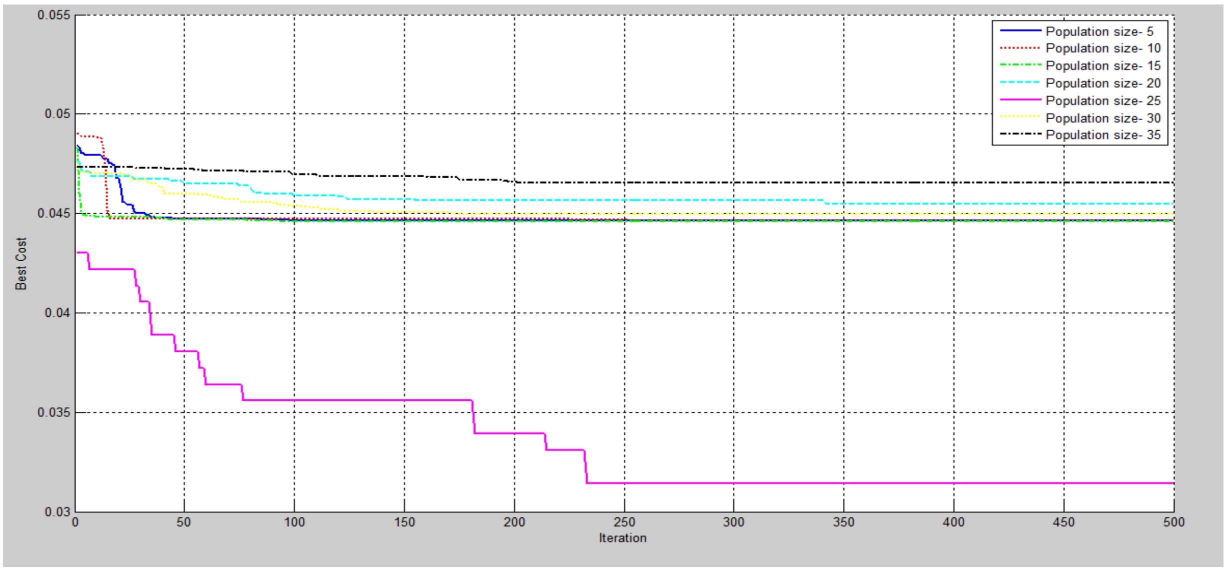


Fig. 6.3 Best Costs for Generations during PSO Pre-training with Different Population Size

Similarly, the best cost recorded during the pretraining of DBN models using SGPSO with various population sizes is shown in Fig. 6.4. It is inferred from the curves that the best cost achieved for models when trained by increasing the population sizes is improving much better with population size and has shown degradation in the case of population size, 35. The models pretrained using population of size 25 and 30 has turned out with competing best cost curves.

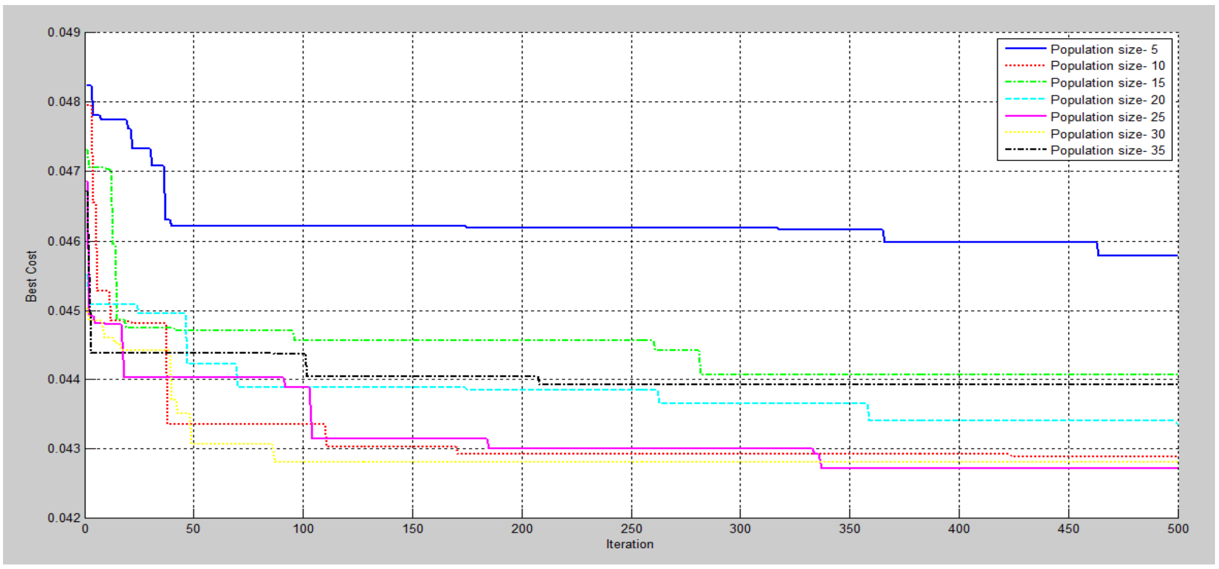


Fig. 6.4 Best Costs for Generations during SGPSO Pre-training with Different Population Size

The best cost curves while pretraining DBN models with NMPSO with various population sizes 20, 30, 50 and 100 respectively shown in Fig. 6.5. The population size used here differs from the other two PSOs due to the existence and implementation of neighborhood relationships.

It can be observed from the best cost curves that the learning happens only for first few generations while using SGPSO for pretraining DBN with the DWTFS dataset.

It is observed for most swarm sizes of PSO pre-training the best costs curve moves more steep for around 50 generations and then stays in the plateau, whereas for most swarm sizes of SGPSO and NMPSO pre-training the best costs curve moves steep by around 100 and 10 generations respectively which then starts to stabilize. This shows that using PSO for pre-training reaches the optimal area in the solution space earlier and then improves steadily in case of swarm size 25 when compared to others. At the same time, SGPSO and NMPSO pre-training show little or no learning.

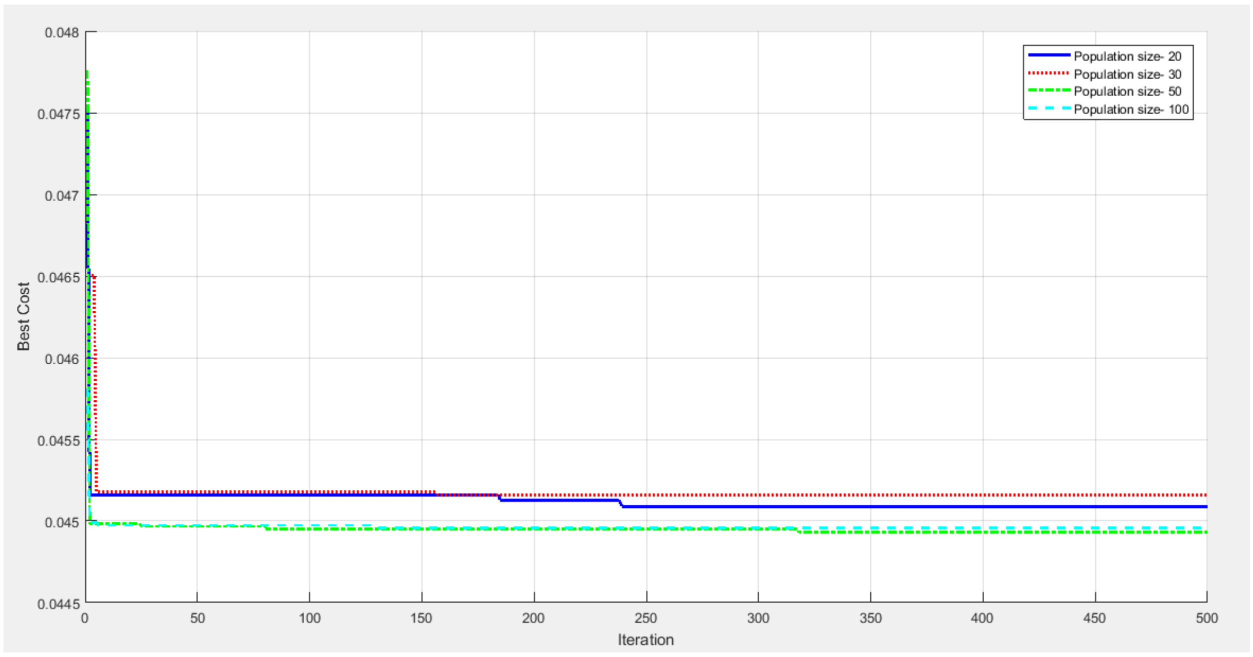


Fig. 6.5 Best Costs for Generations during NMPSO Pre-training with Different Population Size

The various DBN models pretrained using various PSO, are then trained with the backpropagation technique. The hyperparameters DBN are set as follows. The number of layers in DBN is set as 7, the number of neurons in each layers is set as 90x100x120x120x100x70x39, and activation function as sigmoid. The back propagation training is accomplished for 1000 epochs for batch size of 100 data points, step size 0.1, initial momentum 0.5, final momentum 0.9 and weight cost 0.0002. Various performance metrics RMSE, PER, precision, recall, F-measure and accuracy used to evaluate the efficiency of three models. The RMSE values of the DBN models experimented is listed out in Table XIX and Table XX. The observations reveals the increase in population size during pre-training DBNs with PSO, SGPSO and NMPSO have some improvement in RMSE values and PERs by better optimising the initial values assigned to DBN parameters.

Table XIX Average RMSE of DBN Models when Pre-trained with Various Swarm Size

Model		PSO-DBN	SGPSO-DBN	NMPSO-DBN
Population Size	**5**	0.01536	0.016909	-
	10	0.015267	0.016262	-
	15	0.015177	0.016308	-
	20	0.015412	0.016208	0.018343
	25	0.015851	0.016543	-
	30	0.014986	0.016337	0.018502
	35	0.01518	0.015945	-
	50	-	-	0.018122
	100	-	-	0.018008

The best RMSE values observed for PSO-DBN is 0.014986 when pretrained with swarm size 30, where as it is 0.015945 and 0.018008 for SGPSO-DBN and NMPSODBN when swarm size was set to 35. The comparison chart showing RMSEs recorded for the models experiments is depicted in Fig. 6.6. The overall RMSE values observed for PSO-DBN seems to be better the the other two.

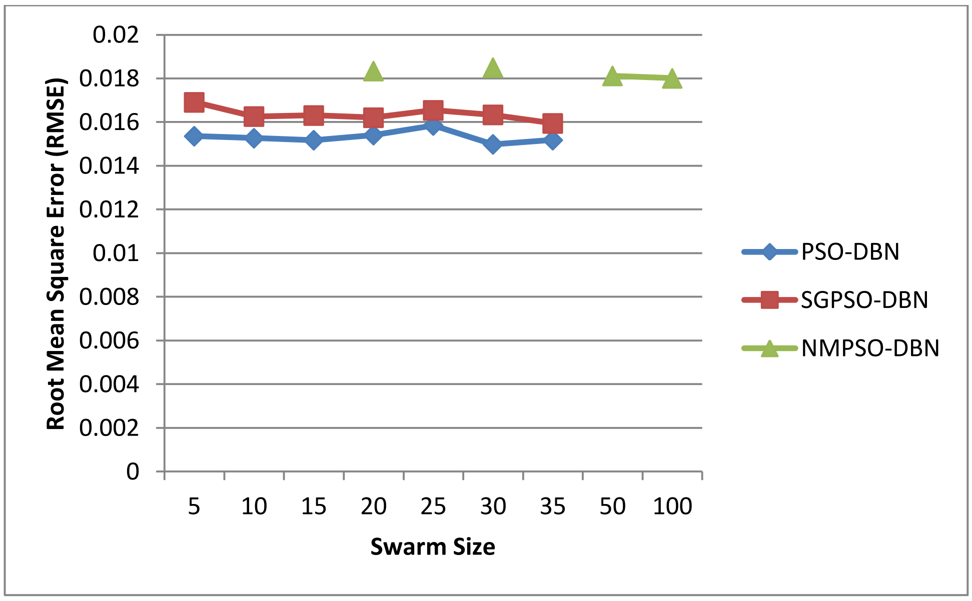


Fig. 6.6 RMSE of DBN Models Pre-trained with PSO, SGPSO and NMPSO with Various Population Size

Table XX Average PER (%) of DBN Models Pre-trained with Various Swarm Size

Model		PSO-DBN	SGPSO-DBN	NMPSO-DBN
Population Size	**5**	12.13	14.15	-
	10	12.02	13.92	-
	15	11.98	13.91	-
	20	12.29	13.83	21.65
	25	12.42	14.1	-
	30	11.39	14.2	19.25
	35	11.99	13.62	-
	50	-	-	19.02
	100	-	-	18.88

The best phoneme error rate achieved amongst various experiments is 11.39% for PSO-DBN pre-trained with PSO population size 30, 14.1% for SGPSO-DBN pre-trained with SGPSO population size 25 and 18.88% for NMPSO-DBN built with swarm size 100. From the Fig. 6.7, the overall PER observed for PSO-DBN shows that it outperforms the other two models for all DBNs built with varying the swarm size that are experimented in this chapter.

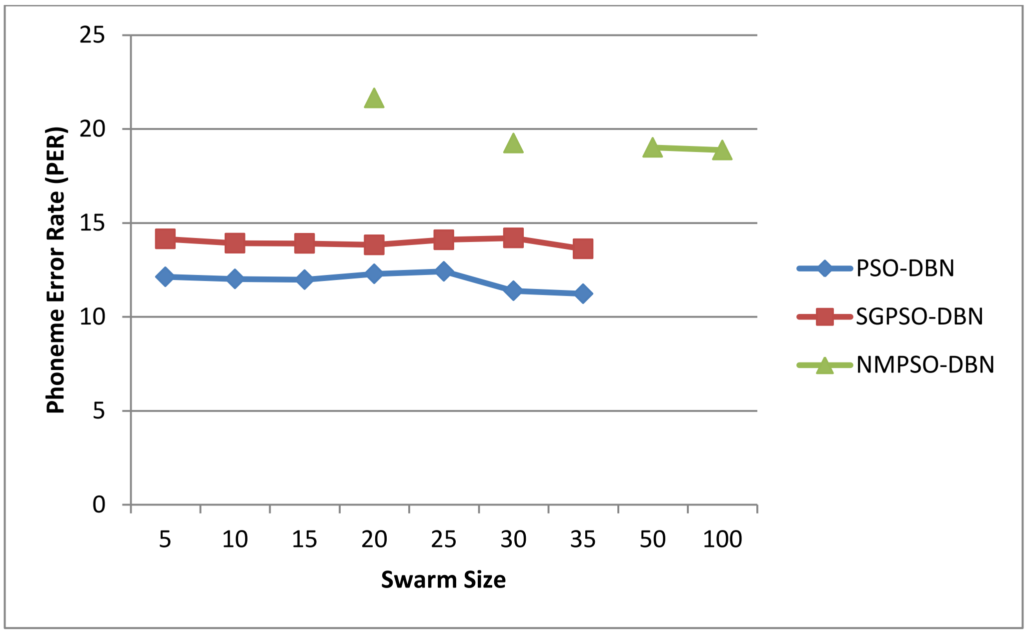


Fig. 6.7 PER of DBN Models Pre-trained with PSO, SGPSO and NMPSO with Various Population Size

The influences of increasing the number of layers for DBNs have been experimented, whose results are discussed here. The RMSE evaluated for training and testing datasets on all the three DBN models are portrayed in Table XXI. The results show an improvement in the RMSE values with increase in number of layers in the DBN. The best RMSE values recorded for PSO-DBN, SGPSO-DBN and NMPSO-DBN, for the training dataset is 0.015497, 0.011965 and 0.016892 respectively, and has been observed with 7-layer DBNs. Similarly, the best RMSE values for corresponding models when subjected with the testing dataset are 0.016204, 0.021122 and 0.020112 respectively.

Table XXI RMSE of PSO-DBN, SGPSO-DBN and NMPSO-DBN Models with Various Network Depths

Method	**No. of Layers**				
	3	**4**	**5**	**6**	**7**
PSO-DBN-Train	0.019058	0.01867	0.017554	0.016104	0.015497
SGPSO-DBN-Train	0.01831	0.017067	0.014958	0.013604	0.011965
NMPSO-DBN-Train	0.019449	0.01892	0.018527	0.017001	0.016892
PSO-DBN-Test	0.019118	0.018416	0.017825	0.016179	0.016204
SGPSO-DBN-Test	0.021025	0.021316	0.021181	0.021489	0.021122
NMPSO-DBN-Test	0.022654	0.021005	0.020898	0.020591	0.020112

The Table XXII lists the phoneme error rates observed for various model experimented in this chapter. The best Phoneme Error Rate for PSO-DBN during training and testing is observed as 12.2% and 12.98% respectively with 7 layers DBN where as the best PER with training set on SGPSO-DBN is 9.5% and on NMPSO-DBN is 16.03% and during testing was found to be 21.92% and 21.09% respectively.

Table XXII PER of PSO-DBN, SGPSO-DBN and NMPSO-DBN Models for Various Network Depths

Method	No. of Layers				
	3	4	5	6	7
PSO-DBN-Train	17.15	15.78	14.46	12.68	12.2
SGPSO-DBN-Train	15.15	13.33	11.94	10.34	9.5
NMPSO-DBN-Train	18.33	16.03	17.57	16.93	16.05
PSO-DBN-Test	17.36	15.38	14.83	12.97	12.98
SGPSO-DBN-Test	21.92	23.44	23.18	23.11	22.03
NMPSO-DBN-Test	23.11	21.87	22.98	21.09	22.45

The RMSE of various DBN models developed are visualized in the Fig. 6.8 and their corresponding PERs in Fig. 6.9. It is clear that the variation reflected in RMSE and PER values of SGPSO-DBN and NMPSO-DBN during their respective training and testing reflects them as a overfitted DBN model, whereas the RMSEs and PERs of PSO-DBNs show lesser deviation in both training and testing, reflecting better generalized representation of the phoneme recognition model.

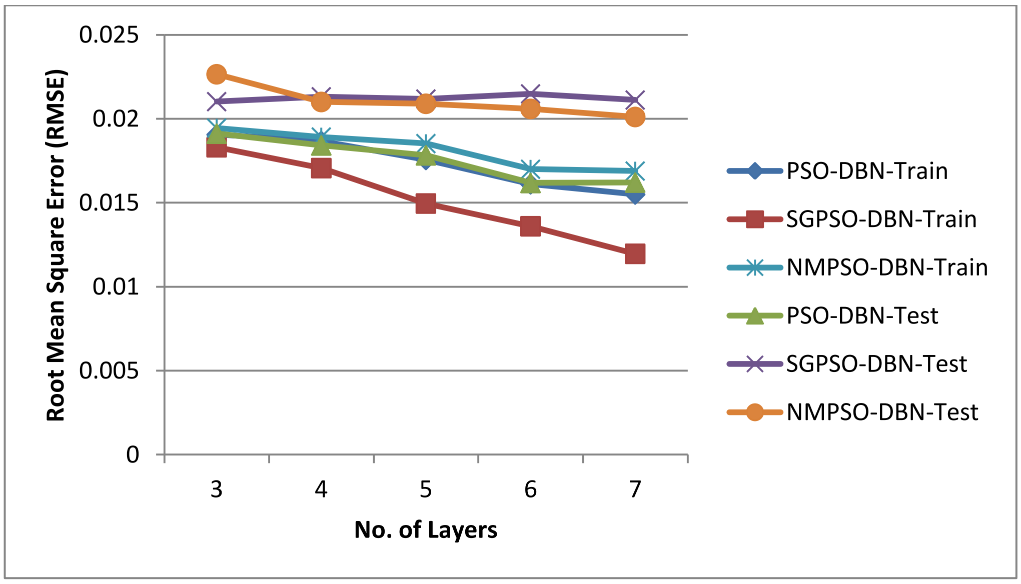


Fig. 6.8 RMSE of Various Models during Training and Testing for Various Network Depths

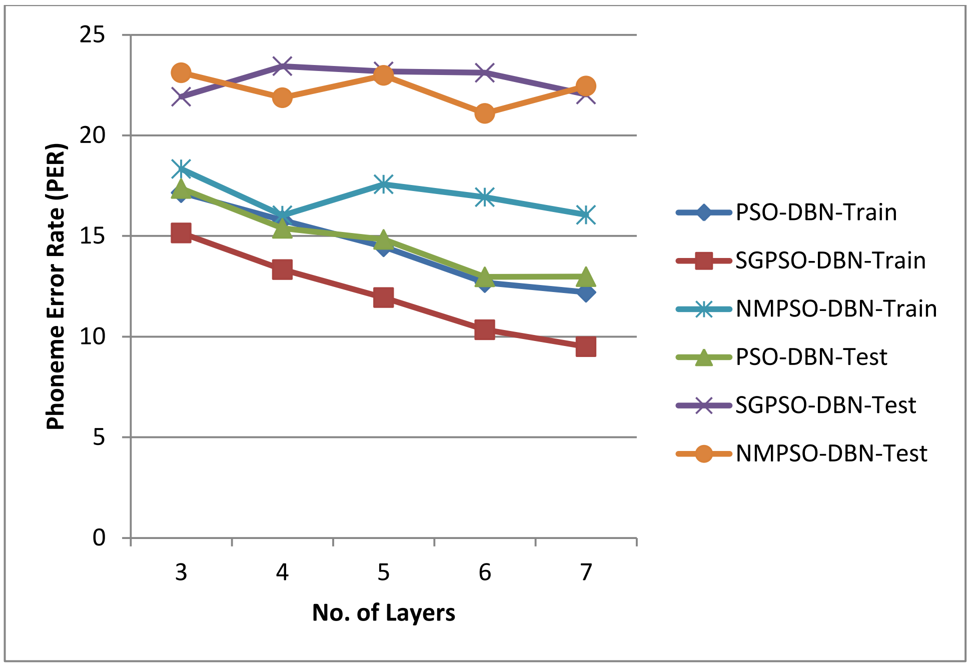


Fig. 6.9 PER of Various Models during Training and Testing for Various Network Depths

The overall performance in terms of precision, recall, F-measure and accuracy are portrayed in Table XXIII for the three models experimented in this chapter. The average precision of PSO-DBN, SGPSO-DBN and NMPSO-DBN are 0.687681, 0.65293 and 0.198742 respectively. The recall of the three models in order is 0.611481, 0.641114 and 0.239211. The F-measure is evaluated as 0.6473, 0.646968 and 0.217105 respectively. The overall accuracy is 87.41% for PSO-DBN and found to outperform the other two models that turned out with accuracies 86.82% and 47.33% respectively.

Table XXIII Performance Metrics of PSO-DBN, SGPSO-DBN and NMPSO-DBN Phoneme Recognition Models

Model	Precision	Recall	F-measure	Accuracy (%)
PSO-DBN	0.687681	0.611481	0.6473	87.41
SGPSO-DBN	0.65293	0.641114	0.646968	86.82
NMPSO-DBN	0.198742	0.239211	0.217105	47.33

The comparison of precision, recall and F-measure is projected in Fig. 6.10, where PSO-DBN shows greater precision, SGPSO-DBN have turned out with greater recall and both PSO-DBN and SGPSO-DBN with equivalent F-measures. The overall accuracy comparison of the three models is illustrated in Fig. 6.11. PSO-DBN phoneme recognition model tops with greater accuracy, followed by SGPSO-DBN and then by NMPSO-DBN models.

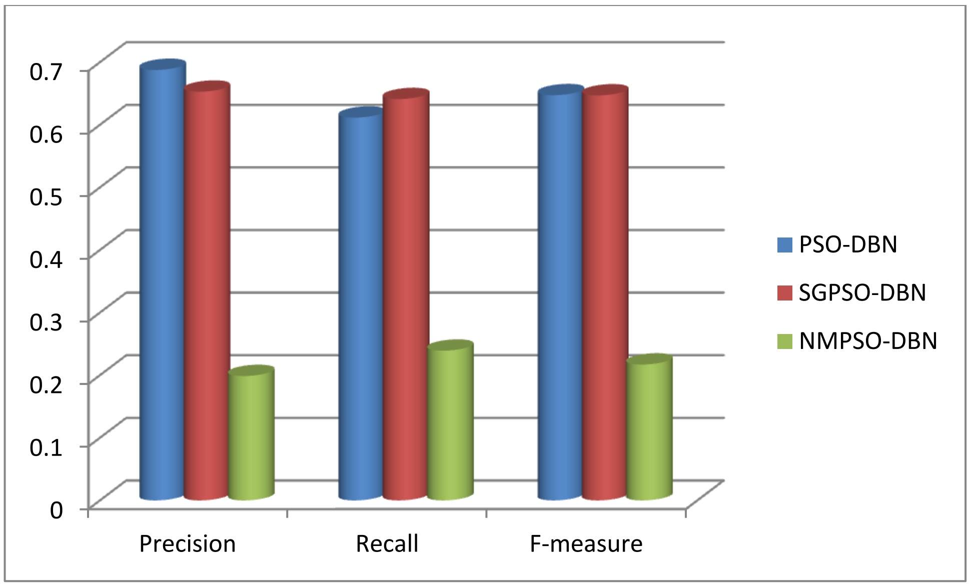


Fig. 6.10 Performance Comparison for the Three Models

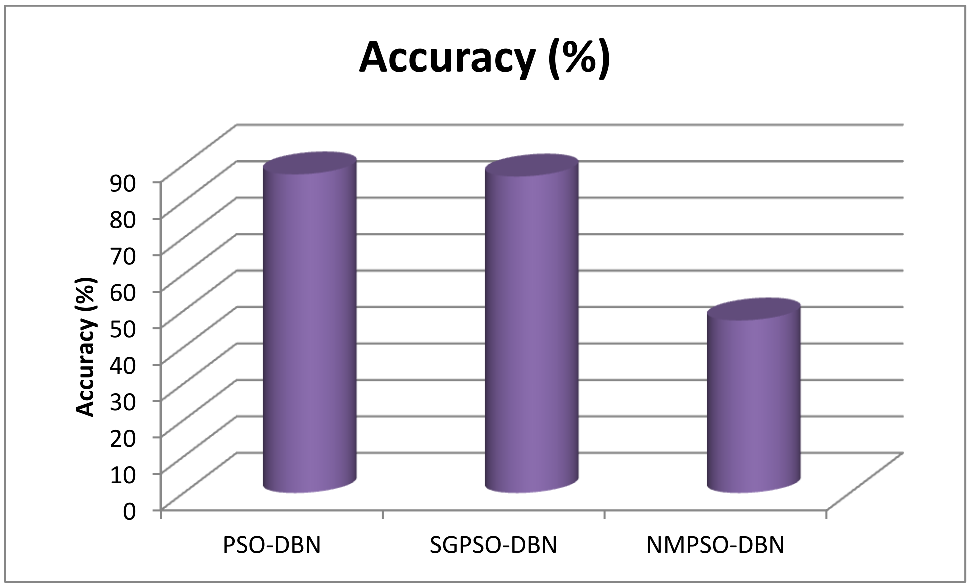


Fig. 6.11 Comparison of Accuracy for the Three Models

In addition to better accuracy, it is obvious that the time complexity of pretraining the DBN as a through PSO and its variants is O(n) for n epochs in training, which is incomparably better to pretraining DBN in isolation of layers with O(mn), where m is the number of layers and n is the number of epochs. The average time taken by the various 7-layer DBN models built in this chapter is compared with the time taken by the 7-layer DBN pretrained with contrastive divergence technique CD-DBN and the comparison is illustrated in Fig. 6.12. The PSO variant incomparably outperforms the contrastive divergence technique based pretraining, among which the basic PSO produces better results.

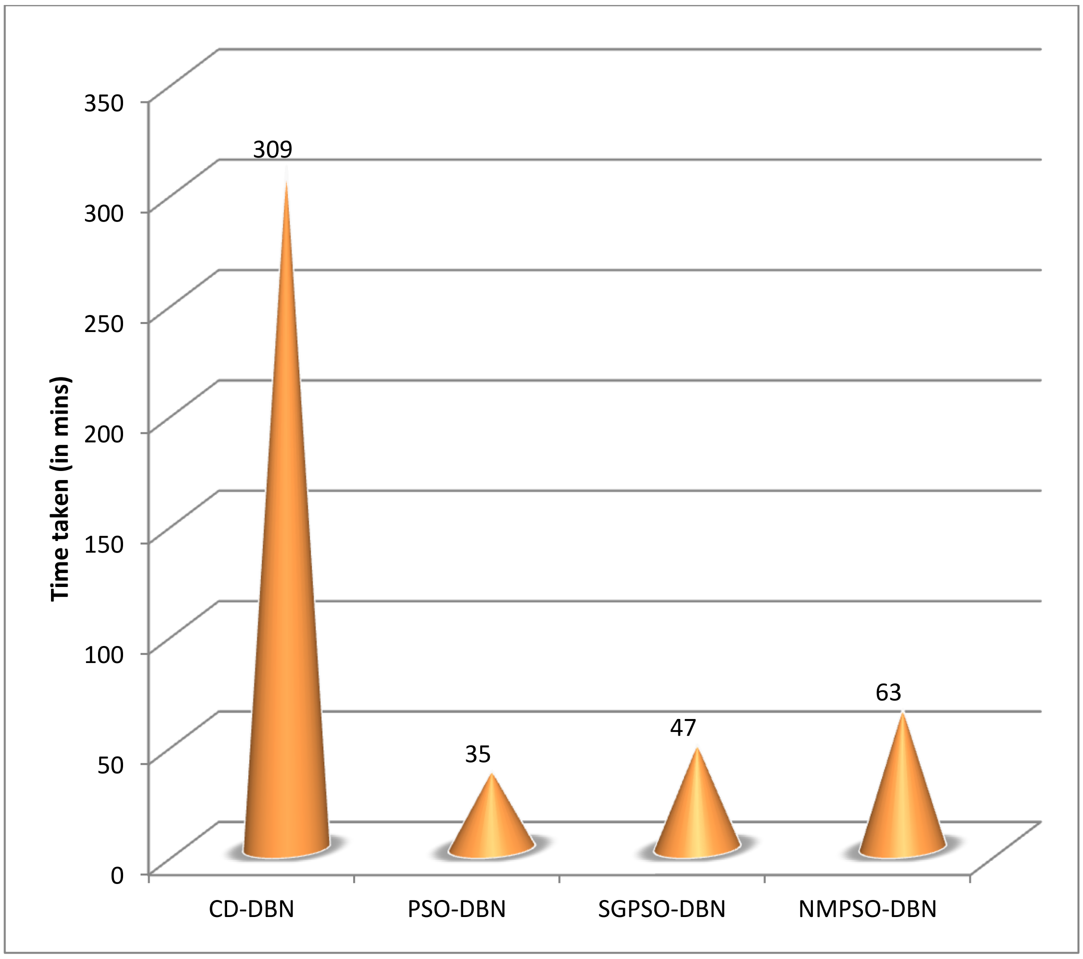


Fig. 6.12 Comparison of Time-taken to Build the Models CD-DBN, PSO-DBN, SGPSO-DBN and NMPSO-DBN

Findings

From the experiments, it is proved that the phoneme classification DBN models built with all the PSO variants in the pretraining phase produces better solutions. The solutions are found to improve when increasing the swarm size to some extent until it reaches certain turning point of swarm size. The turning point defining the swarm size is observed to vary for different models. It is found that the PSO and SGPSO pretraining options provide promising results than NMPSO. The fine tuned DBN with basic PSO outperforms the other two models and also ANN, ANFIS and CD-DBN based acoustic models. Both theoretically and practically, it is evident that replacing contrastive divergence pretraining with PSO variants speeds up the model building process to a greater extent and help promoting to build the models with better optimized parameters by identifying the global optimum in the solution space. But still the risk of stagnation of particles is observed through the best cost curves during pretraining.

SUMMARY

This chapter discussed the development of DBN models using various pretraining procedures namely PSO, SGPSO and NMPSO. The various experiments conducted on building DBN based acoustic model for phoneme recognition are demonstrated. The performances of the various models built by varying the swarm size and the number of layers in the DBN was compared and reported in this chapter. Eventhough it was found that the PSO and SGPSO pretrained DBNs outperform the CD-DBN, there is need to improve the accuracy of the phoneme recognition model and to handle the particle stagnation problem. These problems will be dealt in the forthcoming chapter by proposing temperature controlled PSO to pretrain the DBN.

Remarks

- The article titled ‘Building Acoustic Model for Phoneme Recognition using PSO-DBN’ is accepted to be published in International Journal of Business Intelligence and Data Mining, ISSN online: 1743-8195, ISSN print: 1743-8187. Inderscience. DOI: 10.1504/IJBIDM.2018.10010711 (Accepted-Scopus Indexed)