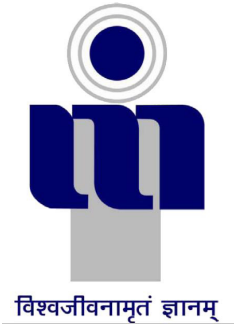
IMPROVISED NATURE INSPIRED DEEP BELIEF NETWORK AND ITS APPLICATION IN HEART DISEASE PREDICTION

A project report submitted in partial fulfillment of the requirements for B.Tech. Project

B.Tech.

by

Kanika Singhal(IPG2013-056)
Ritu Jha (IPG2013-092)
Sarthak Joshi (IPG2013-096)



ABV-INDIAN INSTITUTE OF INFORMATION TECHNOLOGY AND MANAGEMENT
GWALIOR-474 010

MAY-SEPTEMBER 2016

CANDIDATES DECLARATION

We hereby certify that the work, which is being presented in the thesis, entitled **Improvised nature inspired Deep belief Network and its application in heart disease prediction**, in partial fulfillment of the requirement for major project of **B.Tech** and submitted to the institution is an authentic record of our own work carried out during the period *May 2016* to *September 2016* under the supervision of **Prof. Anupam Shukla and Dr.Ritu Tiwari**. We have also cited the reference about the text(s)/ figure(s)/ table(s) from where they have been taken.

Date: Signatures of the Candidates

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Date: Signatures of the Research Supervisors

ABSTRACT

Deep Belief network is a graphical model which learns from a very large data set in a compact manner.Though the function is meticulous but due to high number of input parameters it requires, it become very difficult to train and find the best possible optimized model for it. In this project, integration of deep belief network with a nature inspired algorithm called particle swarm optimisation is done to find the best suited dimension of the RBM layer and the perceptron layer. Further, to test the proposed strategy, standard UCI Hungarian and Cleveland data sets is used.

Keywords: Neural network, Deep belief network (DBN) , Restricted Boltzmann machine (RBM) and Particle swarm Optimization(PSO), UCI ,Deep learning

ACKNOWLEDGEMENTS

We are exceedingly obliged to **Prof. Anupam Shukla** and **Dr. Ritu Tiwari** for giving us the autonomy of working and trying different things with thoughts. We would like to take this chance to express our significant appreciation to them for his scholastic direction as well as for their enthusiasm for our undertaking and consistent backing combined with boosting our confidence and motivating sessions which were exceptionally productive and instrumental in building confidence and trust inside us. We would also like to thank our Director sir **Prof. S.G. Deshmukh** for his valuable support.The nurturing of the present work is essentially because of their important direction, recommendations, adroit judgement, productive feedback and an eye for flawlessness. Our mentors constantly addressed myriad questions with benevolence and exceptional persistence and patience, never giving us a chance to feel that we are beginners by continually listening attentively to our perspectives, appreciating and improving them and by giving us a free hand in the project. It is simply because of their overwhelming interest and supportive demeanor, the present work has accomplished the stage it has.
We are thankful to our colleagues whose constant encouragement served to renew spirit, refocus attention and energy and helped us in carrying out this work.Finally we are grateful to our institution for providing the infrastructure support.

(Kanika Singhal)
(Ritu Jha)
(Sarthak Joshi)

Contents

List of Tables

List of Figures

ABBREVIATIONS

DBN	Deep belief network
PSO	Particle Swarm Optimization
ANN	Artificial Neural Network
UCI	University of California Irvine
MNIST	Mixed National Institute of Standards and Technology
ACO	Ant Colony optimization
NN	Neural Network
FDSS	Fuzzy Decision Support System
MLP	Multi Layer Perceptron
K-NN	K Nearest Neighbour
NN	Neural Network

NOTATIONS

sigm()	sigmoid function
E()	energy function of RBM
epsilon	learning rate
W	weights for edges between a graph
a,b,c	biases for different nodes

Chapter 1

INTRODUCTION AND LITERATURE SURVEY

1.1 INTRODUCTION

In today's world machine learning and data science have a wide scope in the medical field. Heart diseases [Yusuf et al. (2001)] today are an increasing threat to both rich and poor. The lack of physical activity and alcohol are the two main reasons for increasing heart diseases in today's world. The other factors include obesity,smoking, age,pollution and diet also play a vital role in whether a person will suffer from heart disease or not. Death rate due to cardiovascular problem is increasing alarmingly in developing countries and pose a greater threat in comparison to the developed countries. Heart disease mortality in rich countries is less, but it shows a sharp increase in the poor nations [Riley and Cowan (2014)]. Accurate diagnosis of heart diseases is highly important.

Computational methods are used to assist a doctor to diagnose disease of a patient. Classification of medical data requires a high accuracy in predictions so that they can be of precise aid.Medical data set classification and prediction[Soni et al. (2011)] is used in diagnosis of diseases beforehand. Mostly the Doctor and patients are not aware of the disease just on the basis of symptoms ,classification of such a medical data set is the most important problem to solve.

In classification problem there is an input set and output set of class labels and the aim is to find the decision plane which maps them. In medical field, various computer researchers have attempted to apply diverse techniques to improve the accuracy of classification for the given data. Classification techniques whose classification accuracy will give better information to identify the potential patients will therefore be used to improvise the diagnosis accuracy.

In the recent studies, metaheuristic algorithms like simulated annealing, genetic algo-

rithms, and particle swarm optimizations and also data mining techniques like Bayesian networks, artificial neural network,fuzzy logic, and decision tree are applied for classification of medical data and remarkably meaningful results were obtained.
The aim of the project is to make a new kind of classifier algorithm which will correctly predict heart disease and reduce the causalities caused due to cardiac problems.

1.1.1 Machine learning

The art of making the machine learning [Carbonell et al. (1983)] is termed as machine learning. According to Aurther Samuel "Machine learning is a field of study that gives computers the ability to learn without being explicitly programmed".The machine learning is used to solve problems by learning from a given input. The learning is through retaining a good representation of data and creating a generalized algorithm to predict unknown data properly. The various learning algorithms are:

1. **Supervised learning** : This produces a classifier or regression function from labeled data. Both the sets of input and output are available and the goal is to devise a supervised algorithm that maps every input to output. The algorithm learns by minimizing the mean square error averaged over all inputs. The minimization of this error leads to adjustment of weights and next time when algorithm trains it uses these adjusted set of weights. The algorithm is trained by known inputs and their corresponding outputs. When trained on training set inputs, the algorithm gives the desired output. Digit recognition is an example of supervised learning in which input given is handwritten digit and the algorithm is used to classify the type of digit it is [Kala et al. (2010)].

2. **Semi-supervised learning** : It is a type of learning which uses both labeled and unlabeled data .In particular it needs only a small amount of labeled data and a lot of unlabeled data.

3. **Unsupervised learning** : This type of learning is performed solely using unlabeled data.The data is fed into the algorithms and the data is grouped into different clusters using the data distribution. The number of groups also vary according to the the data distribution.The training unit analyze the internal differencing structure of the data and then groups it .Some of the most common algorithms used for unsupervised learning are K-nearest-neighbour,K-means and many more.

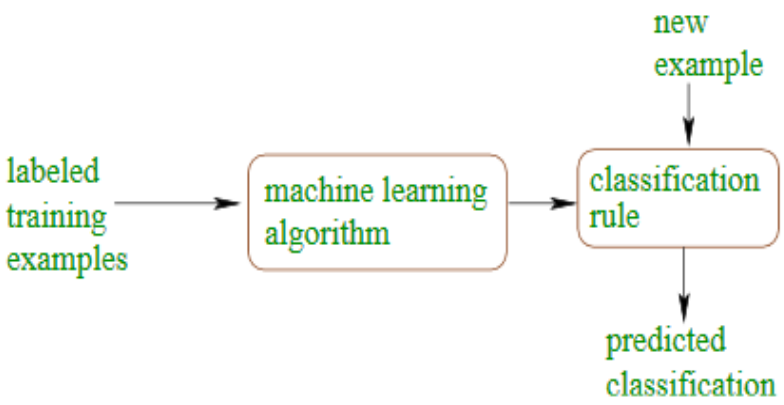


Figure 1.1: Process of Classification

1.1.2 Perceptron

Perceptron [Rosenblatt (1958)]is a mathematical model for supervised learning of binary classifiers.It is one of the simplest neural network.The network takes a vector as input and learns the associated weights, in order to calculate a single scalar quantity for making decisions. If the quantity of decision unit is above some threshold, then the input vector is classified as the target class. Therefore, the perceptron is the algorithm for supervised classification. The learning algorithm is adapting weights by minimizing the error between the target output and the actual output. If the data is linearly separable, then the learning algorithm will converge. This simple network has many limitations, such that for very large data sets it can take even months to get trained.

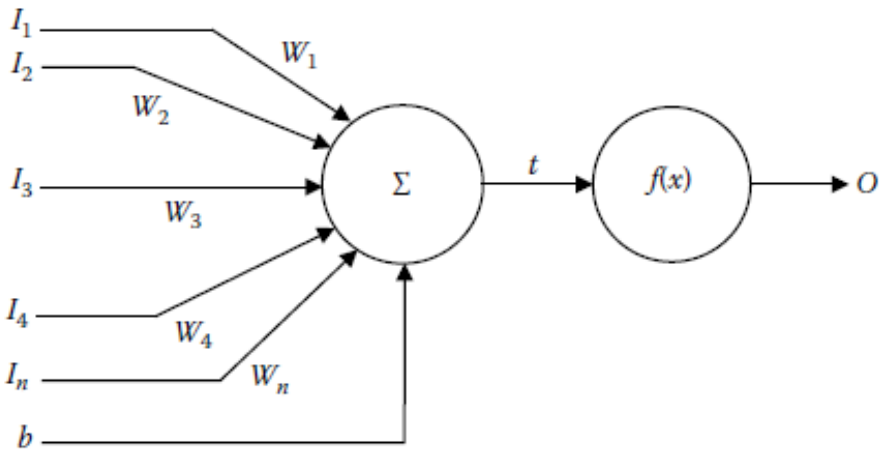


Figure 1.2: The processing in a single artificial neuron [Shukla et al. (2010)]

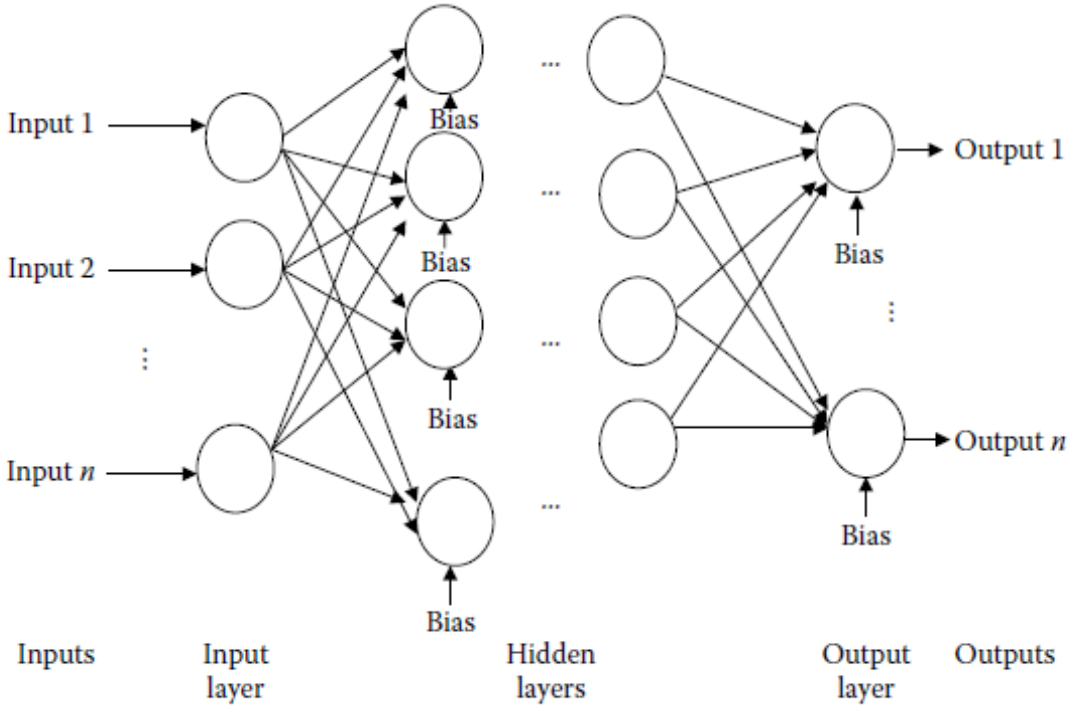


Figure 1.3: The general architecture of the multilayer perceptron [Shukla et al. (2010)]

1.1.3 Artificial neural network

Artificial neural network [Hagan et al. (1996)] resembles biological network of brain neurons.Similar to central nervous systems, neural network consists of an interconnected group of neurons. Each node gets input from other nodes and weights between nodes adjusts so that the whole network learns to compute the output. There are various types of neural networks structures with each having its own learning algorithm. ANN is a novel computer architecture compared to traditional computers. An artificial network of neurons connected with each other to give a specified output on applying input is called neural network.It consists of input layer, various hidden layers and output layer.The characteristics of ANN are:

- Architecture
  It specifies the number of hidden layers, no of neurons, no of interconnections and the weights of the edges.

- Activity Rule
  The activation function deciding the output of neuron is activity rule.
  Neural network weights need to be changed to bring the correct output.This is done with the help of neural network.

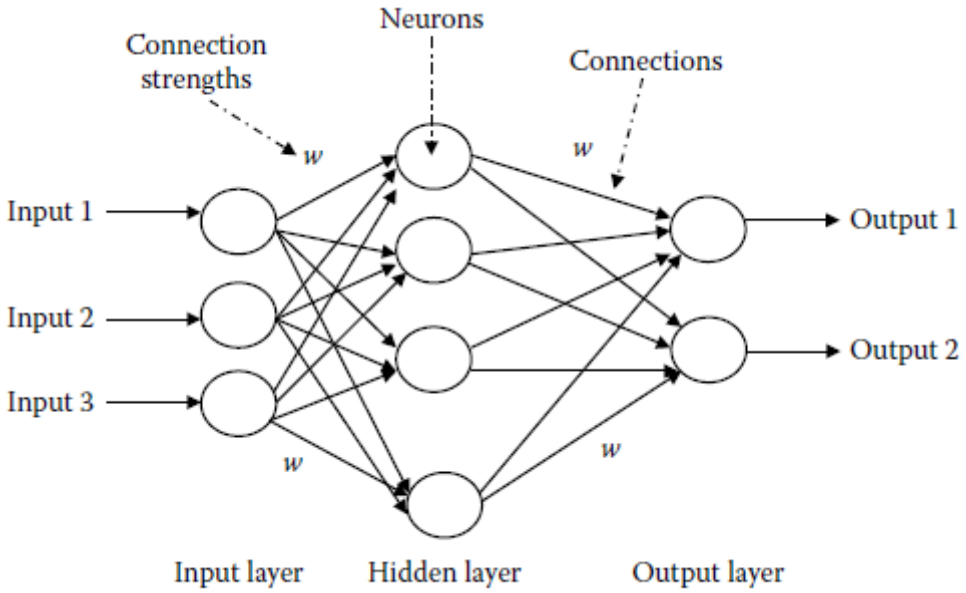


Figure 1.4: Structure of Back propagation neural network Architecture [Shukla et al. (2010)]

1.1.4 Deep learning

In Deep learning [Bengio (2009)] we train a data through a multiple number of neuron layers. With increment in the number of hidden layers the complexity of the hypothesis function increases and so does it become much more difficult to train a deep neural network.Generally in most of the problems a single hidden layer network is enough to give good results but for some complex problems multiple layers are required. We have further in this report proposed a greedy layer-wise method for training deep network.

1.1.5 Restricted Boltzmann machine

Boltzmann machine are bidirectional graphical units that are used to model the probability distribution. Restricted Boltzmann Machine is a type of Boltzmann machine with a restriction that the graph formed between the nodes must be bipartite that is it is the set of vertices that can be divided into two disjoint sets. RBM consists of a hidden layer and a visible layer with weight distribution between them. Here, We define energy distribution function E(v,h) [Bengio et al. (2007)] for the network which is given by the equation 1.1.

$$\mathbf{E}(\mathbf{v}, \mathbf{h}) = \mathbf{h}'\mathbf{W}\mathbf{v} + \mathbf{b}'\mathbf{v} + \mathbf{c}'\mathbf{h} \tag{1.1}$$

Here, W is the RBM weight matrix, b is of the biases of visible layer and c denotes the biases of the hidden layer & h and v denote the hidden and visible layer activations respectively.
The joint probability distribution for the RBM layer is given in equation 1.2 as follows:

$$\mathbf{p(v,h)} = \frac{\mathbf{1}}{\mathbf{N}}\mathbf{e}^{\mathbf{E(v,h)}} \tag{1.2}$$

Here, N in the equation above denotes the normalisation constant for the given distribution. The conditional probability distributions [Bengio et al. (2007)] associated with

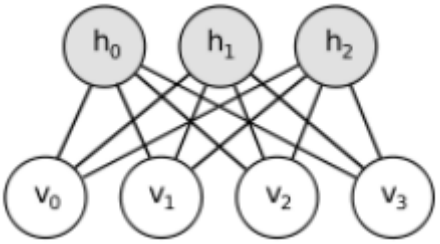


Figure 1.5: Structure of RBM
[Kim (n.d.)]

equation 1.2 is given by Q(h|v) and P(v|h) where Q denotes the conditional probability distribution of the hidden layer activations given the visible layer activations and and P denotes the conditional probability distribution of the visible layer activations given the hidden layer activations.
The mathematical value of Q and P is given below in equation 1.3 and 1.4 respectively.

$$P(v_k = 1|h) = sigm(-b_k - \sum_j (W_{jk} * h_j)) \tag{1.3}$$

and

$$Q(h_j = 1|k) = sigm(-c_j - \sum_k (W_{jk} * v_k)) \tag{1.4}$$

Further, Gibbs sampling is carried out on RBM. In Gibbs sampling process, h is sampled given v and then v is sampled using h. This process in carried till a stopping criterion is satisfied.
Mathematically,
If v_0 is the input observation for the RBM, the log likelihood of v_0 under the model of RBM which is given by equation 1.5.

$$log(P(v_0)) = log(\sum_h P(v_0, h)) = log(\sum_h (e^- E(v_0, h))) - log(\sum_{h,v} (e^- E(v, h))) \tag{1.5}$$

If θ denote the set (W,b,c) that is the collection of variables with respect to RBM. Then the derivative of equation(1.5) with respect to θ is given below. Here, h_0 denotes a sam-

$$\frac{\partial \log P(\mathbf{v}_0)}{\partial \theta} = -\sum_{\mathbf{h}_0} Q(\mathbf{h}_0|\mathbf{v}_0)\frac{\partial \text{energy}(\mathbf{v}_0, \mathbf{h}_0)}{\partial \theta} + \sum_{\mathbf{v},\mathbf{h}} P(\mathbf{v}_k, \mathbf{h}_k)\frac{\partial \text{energy}(\mathbf{v}_k, \mathbf{h}_k)}{\partial \theta}$$

ple of Q($h_0|v_0$) and (v_k,h_k) a sample of the Markov chain. This one step procedure is called Contrastive divergence and its pseudo-code is given in section 2.2.1 as RBMupdate(x,epsilon,w,b,c)function. This function is called repeatedly. The input parameter x is a sample from the training data distribution.

1.1.6 Deep Belief Network

Deep Belief Network (DBN)[Hinton (2011)] is a multi layer belief network. Each layer is a restricted boltzmann machine stacked together to construct DBN. The first step of training DBN is to learn a layer of features from the visible units, using Contrastive Divergence (CD) algorithm. To learn characteristics of features of next hidden layer, previous features activation is treated as visible unit. Finally, the whole DBN is trained when the learning for the final hidden layer is achieved. This simple greedy learning algorithm works for training DBN. This is because that training RBM using CD algorithm for each layer looks for the local optimum and the next stacked RBM layer takes those optimally trained values and again look for the local optimum. At the end of this procedure, it is likely to get the global optimum as each layer consistently trained to get the optimum value.

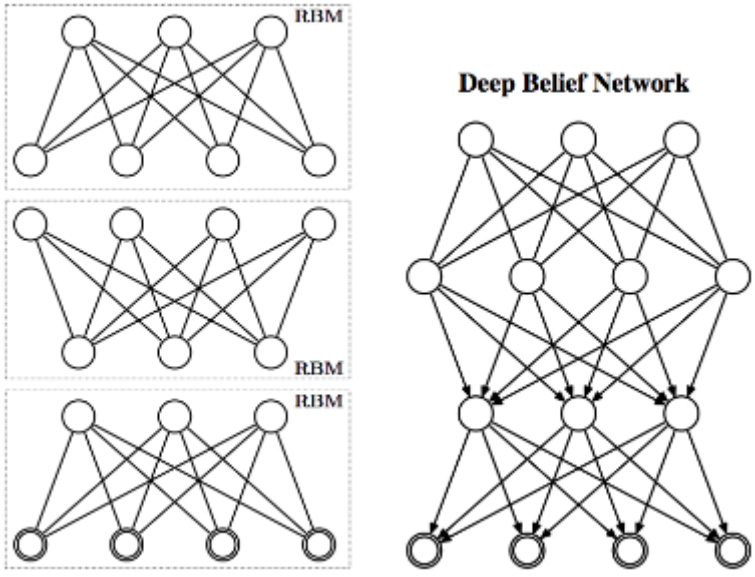


Figure 1.6: Structure of deep belief network containing many RBM layers

The training steps of deep belief network is given by:

1. X is a input feature matrix.
2. Training of RBM on X to obtain is done to obtain its weight matrix W. This weight matrix is used between the two lower layers of the network.
3. X is transformed by RBM and a new data X' is formed .This is done either by doing sampling or by computation of the mean activation of the hidden units.
4. Repeating this procedure with X $\leftarrow$ X' for the next pair of layers. This is done till the two top layers of network are reached.
5. Fine-tuning of all the parameters of this deep architecture with respect to proxy for the DBN log likelihood or with supervised training.

1.1.7 Particle Swarm Optimization

Nature provides examples of optimization problems like a flock of bird arranging themselves for minimization of drag and many more.The taxonomy of the **nature inspired algorithms** can be done into two branches namely evolutionary and swarm optimisation algorithms. Further evolutionary algorithms can be of two types namely genetic and differential evolutionary. Swarm optimisation is also of two types namely particle based (Particle Swarm Optimisation) and colony based (Ant Colony Optimisation).
PSO was introduced by James Kennedy and Russel Eberhart in 1995 [Eberhart et al. (1995)]. It is used to optimise a non-linear function. It is based on information sharing between the agents in the search space. The agents distributed over the function, on the basis of their experience, search for the most optimal solution by sharing information among themselves. Initially all the swarm particles are randomly placed over the space. Each particle calculates its fitness value, then on the basis of the its personal best fitness value and the global best fitness value it update its position and tend to move in the direction of the global best position until the stopping criterion is met.The stopping criterion can be either be if the step size taken by the particles from previous global best to the current global best is less than a particular threshold or it can also be a given number of iterations for which a particle calculates its fitness value and updates its position.

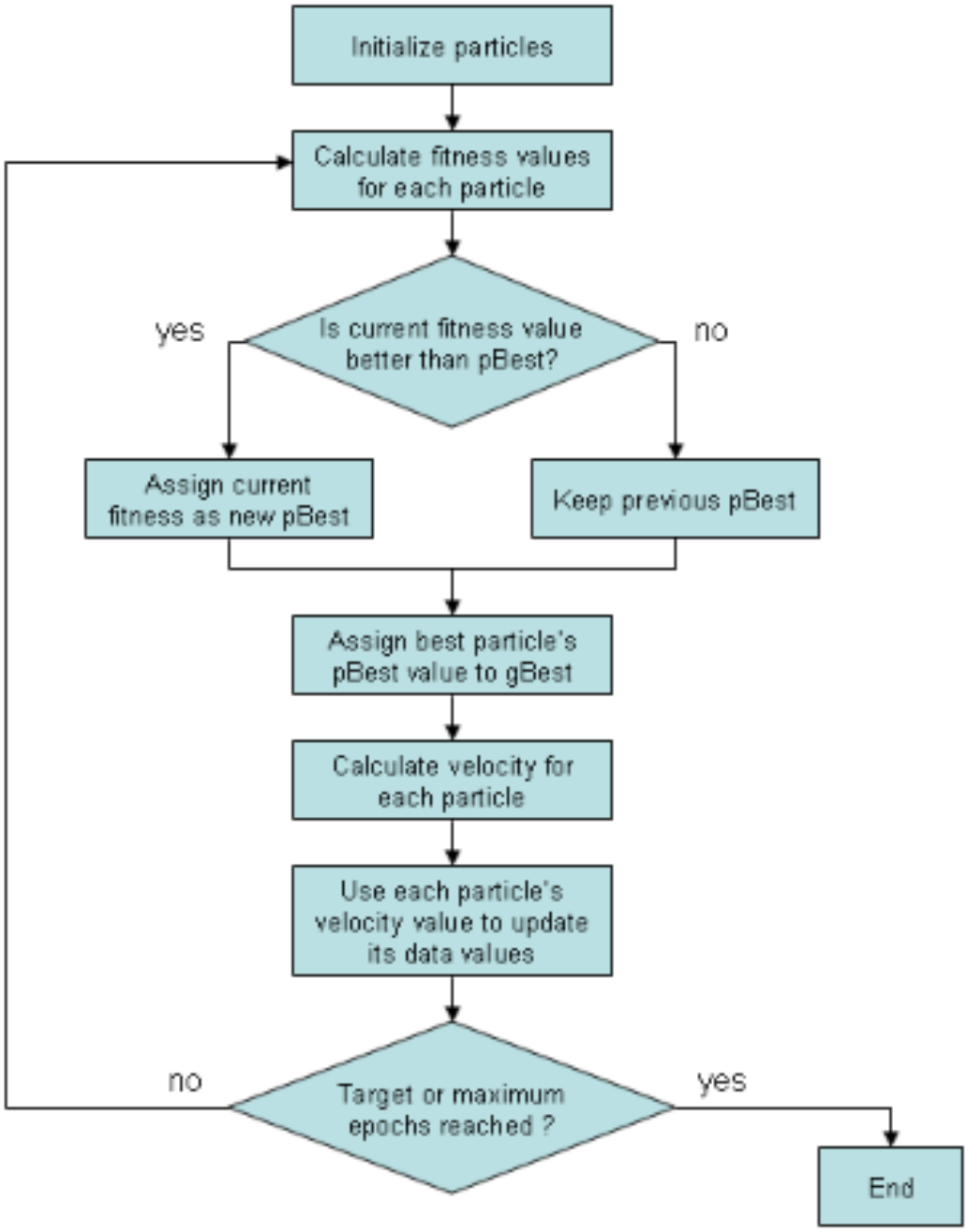


Figure 1.7: Flowchart depicting working of PSO

Particle swarm optimization is similar to genetic algorithm. It is based on the idea of population of particles of potential solutions.The basic idea is that [Shi et al. (2007)]the particles move in continuous space and try to follow the currently best particle. The factors on which speed of particle depends are speed fator due to best local ,speed fator due to best local, weighted influences and speed factor due to inertia. Graphically it is shown in figure 1.8.

Mathematically, Each particle [Shi et al. (2007)] is a E dimensional vector X_i and has

velocity U_i.Some objective function f (fitness) is given.If P_i : is so far best position of particle i and $P_g = \max_i\{P_i\}$. Then, Evolution equation is given as follows:

$$U_i(k+1) = wU_i(k) + c_1r_1(P_i - X_i(k)) + c_2r_2(P_g - X_i(k)) \tag{1.6}$$

$$X_i(k+1) = X_i(k) + U_i(k+1) \tag{1.7}$$

with $w \in [0,1], c_1 > 0, c_2 > 0$ being constants and r_1, r_2 being random numbers in $[0,1]$. Iteration terminates after some time or after $f(P_g)$ reaches some value.

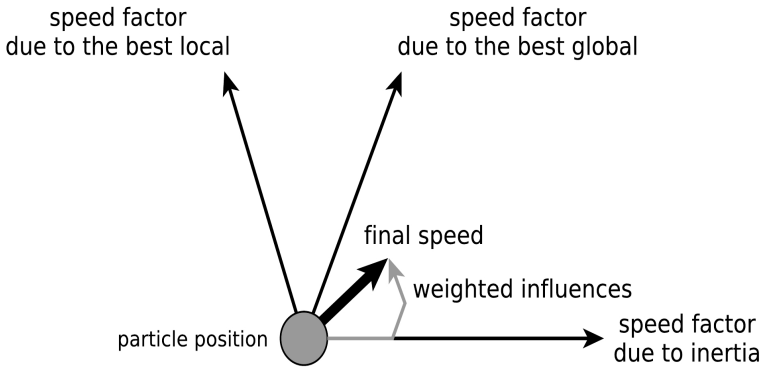


Figure 1.8: Influence of factors on the particle speed in PSO

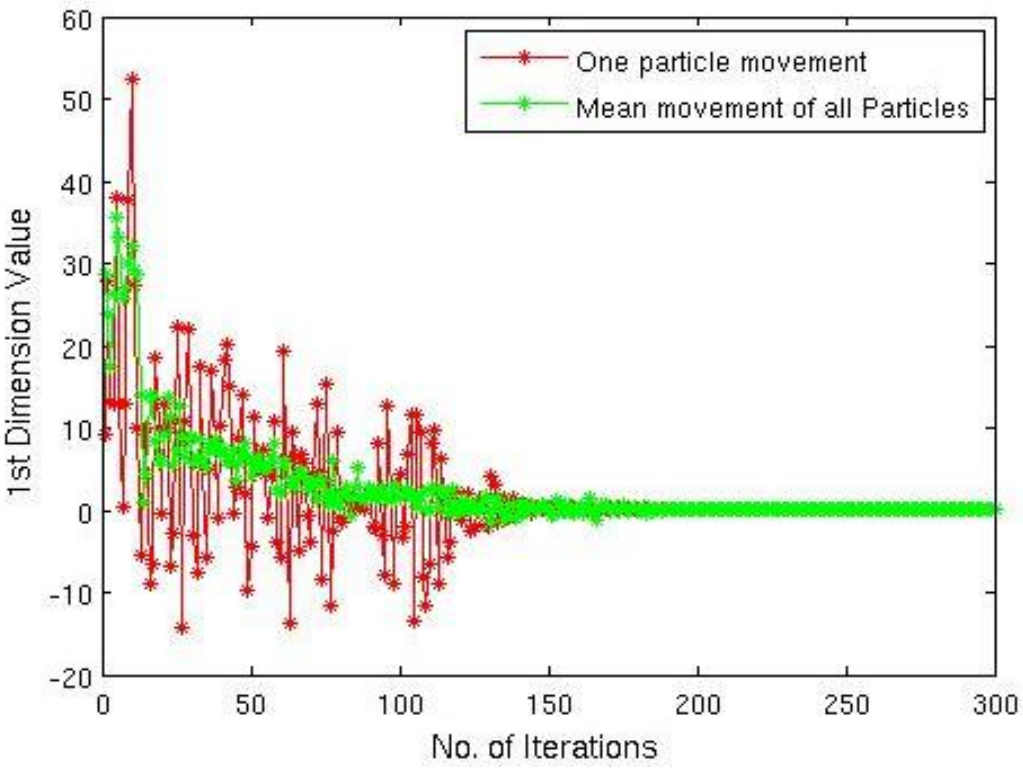


Figure 1.9: Graph between number of iterations and first dimension value

1.2 Problem Statement

On time diagnosis of heart disease is necessary for patient from a cardiologist in an ideal situation.In case of emergencies we need some fast and efficient alternatives. Deep Belief Networks though overcome the problems of deep neural network by following a greedy layer-wise training procedure but has many parameters like regularization and dimensions of neuron layer have a non-linear dependency.This makes it difficult to find the correct value of parameters for DBN.Particle swarm Optimisation has been found very effective in optimising non linear functions and a stochastic approach is used in solving optimisation problem.

1.3 Objective

- To predict the heart disease data using a classifier based on deep belief network
- To optimize this classifier using PSO
- Testing of proposed methodology and comparing the result with existing algorithms on the heart data set

1.4 Literature Review

To fulfill our objective, we studied many literature related to heart disease classification , neural network and deep learning . Many previous works done on our current data set are also reviewed. Typical deep learning architectures include DBNs , SAE [Swietojanski et al. (2014)] , CNNs [Schmidhuber (2015)] extract features from raw unlabeled physiological data.Here, we have listed few recent noteworthy contributions in this field of research. We have also reviewed many variants of NN and DBN.
The basis of neural network was laid in the year 1940s by W. Pitts and McCulloch in [McCulloch and Pitts (1943)] McCulloch and Pitts employed logic and the mathematical notion of computation introduced by Alan Turing (1936-37) in terms of what came to be known as Turing Machines to explain how neural mechanisms might realize mental functions.Multilayer perceptron is better than a single perceptron [Gardner and Dorling (1998)] .The following paper [Saeed (n.d.)], gives a high accuracy of 99.32% accuracy with digit data using MLP Neural Network.
As per literature and above discussion, for past decades several classification tools are available for medical dataset classification. Even then, ANNs are widely accepted and utilized to solve the real world classification problems in clinical applications. Artificial neural network perform better than other classifiers because they are capable of generalization, they are also capable of conditioning,they have a low requirement of

training points and they also converge very fast [Sivanandam and Deepa (2007)]. Various learning or training algorithms for several NN architectures have been proposed in various problems in engineering science and technology and even in some parts of business industry and medicine. A few notable classification applications include biomedical medical diagnosis,handwritten digit recognition, pattern recognition, text categorization, information retrieval, and prediction of bankruptcy[Huang et al. (2006)] and [Huang and Chen (2008)] .

[Maind and Wankar (2014)] there are various advantages of ANN over conventional approaches. Due to its ability to classify the non-linear relationships, classification is difficult without ANN. Many different data sets are used with ANN like handwritten digit data set, iris data set etc. [Sakshica and Gupta (n.d.)] In this Hand digits are recognized using three approaches which are BPO, Single Layer Perceptron, HNN. Among the all methods BPA has been found most accurate as weights are to minimize errors with high accuracy. In case of error BPO can achieve a fast convergence and a satisfiable local minimum. The paper concluded with the fact that BPO can be explored more to improve its performance.

Many improvement strategies are applied on ANN like using GPU instead of CPU to improve speed .[Jagtap and Mishra (2014)] This paper presents fast efficient ANN for handwritten digit recognition on GPU to reduce training time with PTM. BPO GPU based parallelization should be preferred generally with compared to CPU program. For smaller dataset, CPU is good but for larger datasets GPU is preferred. In this book ,[Shukla et al. (2010)] Real Life Applications of Soft Computing robotic controller is formed using neuro-fuzzy approach.

In neural network if we make dense network with many layers, computation becomes difficult so a new concept came into light "Deep Learning".This was first given by Geoffrey Hinton [LeCun et al. (2015)]. Compared to non deep architectures deep learning architectures perform better in pattern and speech recognition. This paper [Hinton (2011)] describes deep belief nets as probabilistic generative models that are composed of multiple layers of stochastic latent variables (also called "feature detectors" or "hidden units").

Many algorithms have been implemented on UCI heart data[heart data set (uci)] set, [Lichman (2013)] Cleveland and Hungarian. [Mangasarian and Musicant (2000)] In this paper various algorithm like SVMLight , ASVM and CPLEX are used on Cleveland data which showed a maximum of accuracy for 85.56%.With ANFIS an error of 15% was obtained [Ziasabounchi and Askerzade (2014)]. ANFIS which is used for classification which has both the advantages of neural network and fuzzy logic. [Bhatia et al. (2008)] the proposed method here gives a classification accuracy of 90.57% which is better than the previous results. SVM is used for heart disease classification and to select critical features.[Noor Akhmad et al. (2009)] This paper compares various

other algorithms like MLP-ANN,K-NN,C4.5,Ripper with FDSS which is fuzzy decision support system on both hungarian and cleveland data sets.
We also reviewed some nature inspired algorithm like PSO. [Bai (2010)] Here, PSO is compared with the other algorithms and it was found that it needs fewer parameters.The paper concludes with the fact that more research can be made on PSO by improving the topology of particle swarm, by blending with other intelligent optimization algorithm and by using PSO to optimize other algorithms [Rini et al. (2011)] In this paper, a review on different methods of PSO algorithm is made. The various variants of PSO are discussed here which shows the advantages over basic PSO. [Ribeiro and Schlansker (2003)] This paper shows the optimization of ANN by PSO. PSO overcomes the limitations of ANN since it does not care about what it is optimizing. Any parameters of network can be optimized using PSO.[Kennedy (2011)] The relationships between PSO and both AI and GA are described.PSO is introduced and is used to optimized highly complex non-linear functions which cannot be classified by neural network.Another work [Anooj (2012)] on UCI data set showed accuracy of 53.862 % on cleveland data set using weighted fuzzy technique.In hungarian it gave an accuracy of 50.583%.
PSO is chosen in our proposed strategy to optimize DBN's parameters because of the above mentioned reasons.

1.5 Research gap

(1) DBN optimized with nature inspire algorithm is a completely new way of improving efficiency of DBN and no work has been done in this area till date.

(2) Deep belief network is inefficient because it may imitate complex functions during training and beginning layers are not able to learn much as they get stuck in training.

(3) DBN has many parameters to adjust before it could predict accurately . This consumes time and so ultimately time required for training of algorithm increases.

(4) To save time, if training is not done for sufficient number of iterations, it may give inaccurate results due to incomplete training.

Chapter 2

DESIGN DETAILS AND IMPLEMENTATION

2.1 Data set description

The data sets used for testing the experiments are benchmark data sets taken from UCI machine learning repository[heart data set (uci)]. The Description is as follows:

1. **Cleveland clinic foundation Database**
   This database consist of 303 records of patients. Each record has 14 attributes. The attributes are given in the table 3.1. The output is either 0,1,2,3 or 4. If the output is 1,2,3 or 4 it represents person is not in normal state and has some cardiac disease based on its severity and 0 indicates that person is normal and has no disease.

2. **Hungarian Database**
   This database consist of 294 records. Each record has 14 attributes. The output of each attribute is either 0 or 1. 0 indicates the absence of any disease and 1 indicates the disease is present. [Noor Akhmad et al. (2009)].

Database classes of patients	0	1	2	3	4	Total patients
Cleveland	164	55	36	35	13	303
Hungarian	188	37	26	28	15	294

Figure 2.1: Distribution of patients in datasets

Attribute	Description	Value Description
age	Age	Numerical
sex	Sex	1 if male, 0 if female
cp	Chest pain type	1 to 4
trestbps	Resting systolic blood pressure	Numerical
chol	Serum cholestrol(mg/dl)	Numerical
fbs	Fasting blood sugar	1 if yes and 0 if no
restecg	Resting cardigraphic results	0 to 2
thalach	Maximum heart rate achieved	Numerical
exang	Exercise induced angina	1 if yes and 0 if no
oldpeak	ST depression induced by excercise relative to rest	Numerical
slope	The slope of the peak exercise ST segment	1 upsloping and 2 flat
ca thal	Number of major vessels	Numerical 3,6 or 7
num	Output Attribute	0 or 1

Figure 2.2: Summary of disease attributes [Noor Akhmad et al. (2009)]

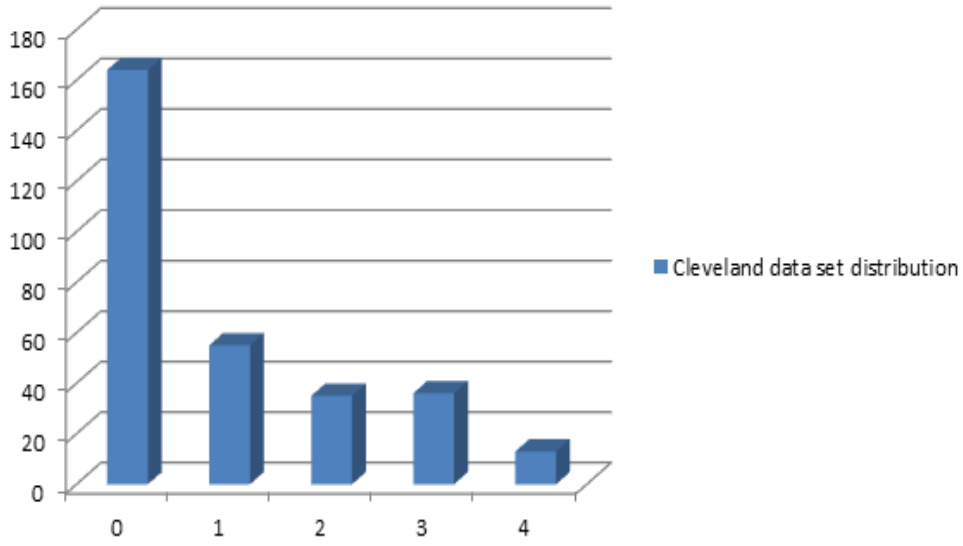


Figure 2.3: Data distribution of Cleveland dataset

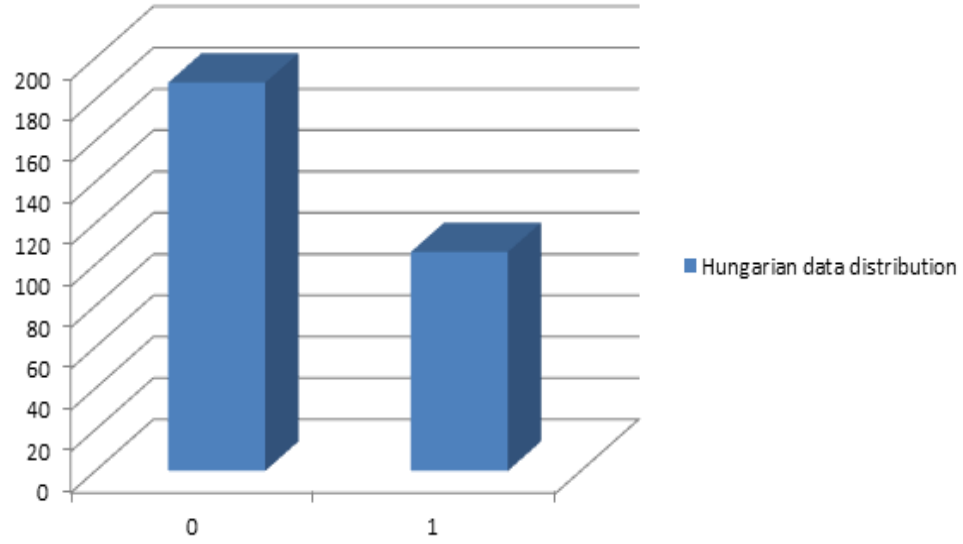


Figure 2.4: Data distribution of Hungarian dataset

2.2 Limitations of existing algorithm

The two existing algorithms considered here are Neural Network and Deep belief network.We give a modified algorithm which is better than these two algorithms.It overcomes the disadvantages of this algorithm.

2.2.1 Neural network limitations

The problems faced by neural network are: [Shukla et al. (2010)]

1. **Convergence**
   Sometimes error functions in ANN may not converge to a particular error point after which it becomes constant. It may increase or decrease abruptly .This is because of wide range of input and output, large learning rate, due to noisy data or some incorrect features chosen to train the network.

2. **Over generalization**
   It may occur where the training algorithm starts to behave like some complex function. This happens when neural network is too much fitted due to training data and give errors on testing with validation data.

3. **Complexity**
   The number of layers and their neurons may increase the complexity of the neural network.

4. **Deep neural networks**
   A network with large number of hidden layers is called deep network. Due to large number of interconnections ,no of neuron layers it is seen that deep networks are not better than non-deep networks. Beginning layers of deep neural network get struck in training without any learning. Different layers learn at different speeds which makes it unfit. Deep neural network finds hard to train with less data and this may result in over fitting.Also in deep nets it is likely that the function will be stuck in local minima and so training with gradient descent no longer works. Gradients become very small and therefore gradient descend does not perform correctly with randomly initialized weights. As the depth of net increases the gradients which were propagated backwards rapidly decrease in magnitude. As a result the derivative in previous layers become low. Thus the earlier layer's weight change slowly.They are not able to learn much.This problem is called the diffusion of gradients.

2.2.2 Deep belief network vs deep neural network

Data is rare, and subsequently for some issues the parameter fitting is hard in complex deep neural network models as we don't get enough data to do so. The question which comes here is how to train deep neural network.Here deep belief network comes into light. This is a greedy layer wise training method. The main idea is to train the layers of the network one at a time, so that we first take 1 hidden layer train it then take 2 hidden layers and so on.The next kth layer takes k-1th layer as input. The training layer's weights individually are then used to initialize the weights in the complete deep network and then the structure entirely is "fine tuned".This means that the structure is trained all together for optimizing the training set error.After we have trained the network using unlabeled data it is more probable that gradient descend will reach global minimum.This is because the unlabeled data provides a good amount of prior info about what the patterns present in input data. This is the basic working in a DBN.

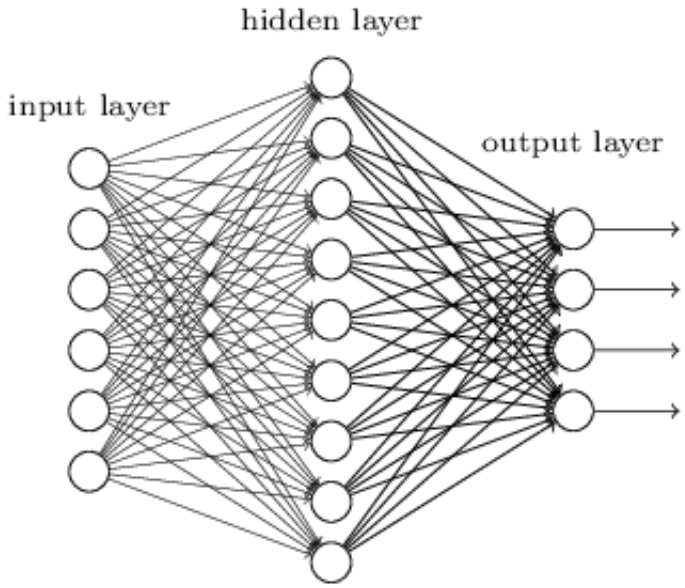


Figure 2.5: Non deep feed-forward neural network

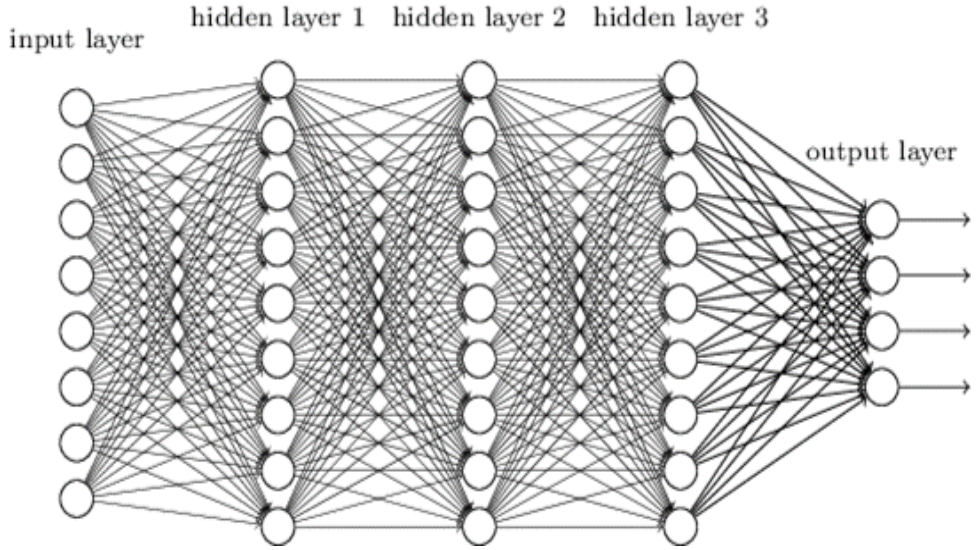


Figure 2.6: Deep neural network

2.2.2.1 Pseudo code of deep belief network

RBMupdate(v[0], epsilon, W, a, b)

1. *for all hidden units i*
2. *compute Q(h[0][i]= 1|v[0])#for binomial units,sigmoid(b[i]+sum_j(W[i][j]*v[0][j]))*
3. *sample h[0][i] from Q(h[0][i] = 1 | v[0])*
4. *for all visible units j:*
5. *compute P(v[1][j] = 1 | h[0]) # for binomial units,sigmoid(c[j] + sum_i(W[i][j]*h[0][i]))*
6. *sample v[1][j] from P(v[1][j] = 1 | h[0])*
7. *for all hidden units i:*
8. *compute Q(h[1][i] =1|v[1])# for binomial units, sigmoid(b[i]+sum_j(W[i][j]*v[1][j]))*
9. *W += epsilon * (h[0] * v[0]' - Q(h[1][.] = 1 | v[1]) * v[1]')*
10. *a += epsilon * (h[0] - Q(h[1][.] = 1 | v[1]))*
11. *b += epsilon * (v[0] - v[1])*

PreTrainSupervisedDBN(Z, epsilon, L, n, W, b, V, c)

```
1.  let X the marginal over the input part of Z
2.  initialize b[0]=0
3.  for l=1 to L-1
4.      initialize W[i]=0, b[i]=0
5.      while not stopping criterion:
6.          sample g[0]=x from X
7.          for i=1 to l-1:
8.              sample g[i] from Q(g[i]|g[i-1])
9.      RBMupdate(g[l-1], epsilon, W[l], b[l], b[l-1])
10. initialize W[L]=0, b[L]=0, V=0, c=0
11. while not stopping criterion:
12.     sample (g[0]=x, y) from Z
13.     for i=1 to L-1:
14.         sample g[i] from Q(g[i]|g[i-1])
15.     RBMupdate((g[L-1],y), epsilon, (W[L],V'), b[L], (b[L-1],c))
```

DBNSupervisedFineTuning(Z, C, epsilon_C, L, n, W, b, V,c)

1. *Recursively define mean-field propagation*
   mu[i](x)=Expectation(g[i] | g[i-1]=mu[i-1](x))
   where mu[0](x)=x,and Expectation(g[i] | g[i-1]=mu[i-1])is the expected value of g[i] under the RBM conditional distribution Q(g[i] | g[i-1])
   when the values of g[i-1] are replaced by the mean-field values mu[i-1](x)
   # In the case where g[i] has binomial units
   # Expectation(g[i][j] | g[i-1]=mu[i-1]) = sigmoid(b[i][j] +
   sum_k W[i][j][k] mu[i-1][k](x)

2. *Define the network output function f(x) = V * mu[L](x)' + c*
   Iteratively minimize the expected value of C(f(x),y)
   for pairs (x,y) sampled from Z by tuning parameters W, b, V, c
   This can be done by stochastic gradient descent with learning rate epsilon_C,
   using appropriate stopping criterion such as early stopping on validation set

TrainSupervisedDBN(Z, C, epsilon_CD, epsilon_C, L, n, W, b, V):

1. *let X the marginal over the input part of Z*
2. *PreTrainUnsupervisedDBN(X, epsilon_CD, L, n, W, b)*
3. *DBNSupervisedFineTuning(Z, C, epsilon_C, L, n, W, b, V, c*

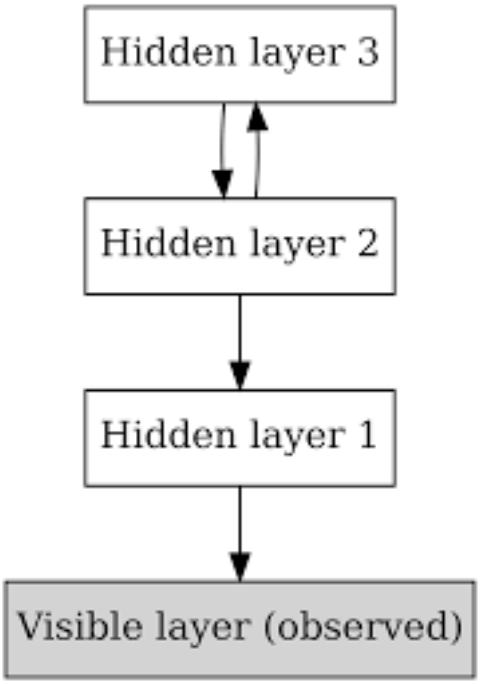


Figure 2.7: Deep belief network structure

2.2.2.2 Limitations of Deep belief Network

(i) It requires a substantial amount of data for training else they overfit.

(ii) There are many hyper parameters to tune like number of layers, number of hidden units, learning rate and many more.The accuracy function is determined by these parameters therefore it is necessary to optimize all these parameters to get optimized results

(iii) To choose the most efficient values of all these input parameters and train the system accordingly its takes a considerable amount of time. Therefore there is a need of some algorithm which can optimize these parameters in limited time giving the most optimized performance.

2.3 Proposed strategy

[Hinton et al. (2006)]In this paper Hinton said that learning becomes quite difficult in very densely-connected belief nets that have many layers because it is hard to infer the conditional distribution of hidden activities data vector. To overcome the observed limitations of the deep belief network, the project proposes an improved strategy. The improved strategy introduced here is DBN optimized by PSO.

2.3.1 Deep belief network with particle swarm optimization

The improvised strategy aims to reach an optimal solution using nature inspired algorithm - Particle swarm optimization. It searches for the best combination of dimensions for DBN and so helps in calculating the dimensions to be used in unsupervised RBM and supervised NN training.The number of layers to be used in RBM and NN layer can be found in a very optimized and cost effective way. Figure 1.9 shows movement of particles in a swarm. In the beginning they are scattered but ultimately the meet and converge to the global minima. Initially movement was random , after few iterations the movement of particles became more directed towards the goal.

Pseudo code for particle swarm optimization is described below.

```
1.  For each particle
2.  {
3.      Initialize particle
4.  }
5.  Do until maximum iterations or minimum error criteria
6.      For each particle
7.      {
8.          Calculate Data fitness value
9.          If the fitness value is better than pBest
10.             {
11.                 Set pBest = current fitness value
12.             }
13.         if pbest is better than gbest
14.             Set gBest = pBest
15.     }
16.     For each particle
17.     {
18.         Calculate particle velocity
19.         Use gbest and Velocity to update particle data
20.     }
```

2.4 Methodology

The flow diagram in figure 2.8 represents these steps in graphical form.
Step 1: Collecting benchmark datasets which contains heart data.
Step 2: : Implementing DBN and other algorithms like NN,K-NN,SVM to check the presence of coronary disease.
Step 3: : Examining the data and comparison of results obtained using statistical means and deducing the best algorithm.The best algorithm obtained is DBN it gives highest accuracy among all.
Step 4: Finding out limitations of the best existing algorithm.
Step 5: Proposing and implementing an improvement strategy to overcome the limitations and optimizing the results.this improvement strategy includes our hybrid algorithm ,DBN modified with PSO.
Step 6: Analysis of results, findings and documentation.

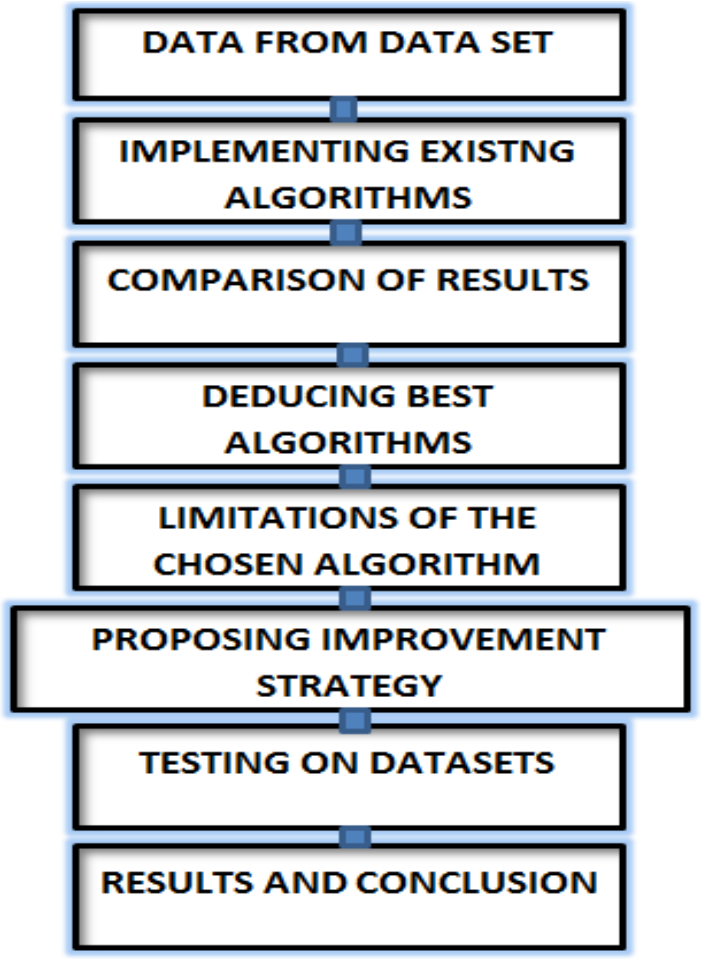


Figure 2.8: Flowchart representing methodology

The flow diagrams in figure2.9 and 2.10 show detailed design of our proposed algorithm.

Step 1: The data collected is normalized and rescaled.

Step 2: This data is fitted into RBM layer of DBN.

Step 3: The RBM does unsupervised training on this data to pass it to the next layers on neural network.

Step 4: This transformed data is passed through NN which does supervised training on data.

Step 5: All the above steps are put into a function and our aim to maximize the function output i.e. accuracy given by the function.This accuracy is the accuracy of our system related to how much efficient it is to predict heart disease

Step 6: This optimization is done by PSO . For that,input all parameters to the function with RBM and NN layer size kept variable.

Step 7: DBN is repeatedly called by PSO.At every step swarm particles move closer towards better accuracy.

Step 8 :The dimensions of RBM and NN which give highest accuracy is given as output by PSO.

Step 9: Print the accuracy and predict result

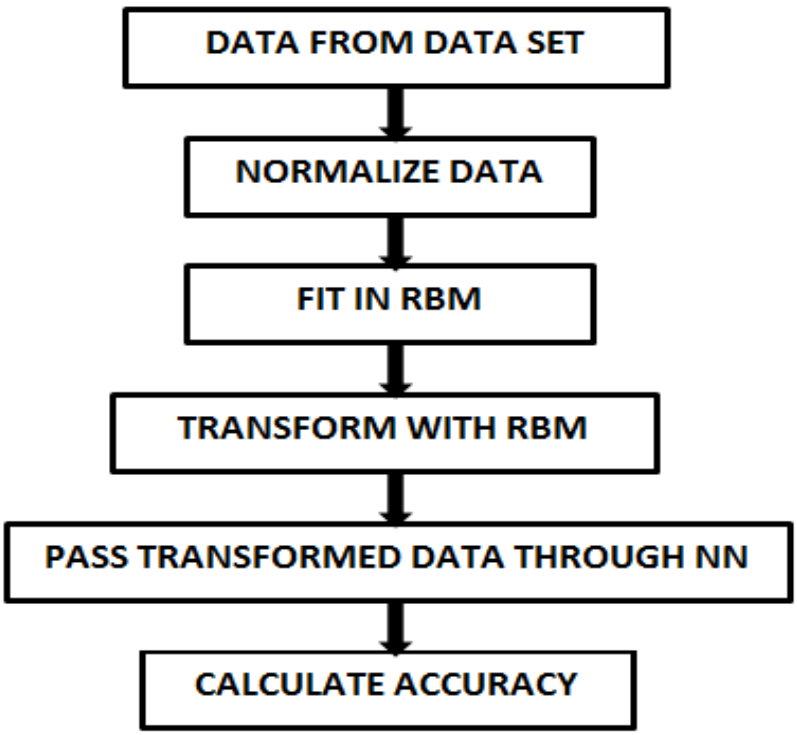


Figure 2.9: Flowchart representing DBN

2.5 Proposed algorithm

A function is made named deep-belief-network is taken as the objective function for optimization of input parameters. The parameters required for the deep belief net are training data ,training labels,neural network layer size, number of iterations for RBM pre training, number of iterations for neural net training, regularization factor for neural network ,learning rate of RBM , learning rate of neural net, batch size for neural network training and batch size for RBM. The two most important factors namely the dimensions of neural network layer (directed layer) and the dimension of RBM layer(undirected layer) are kept as variable to be optimized by particle swarm optimization. The output of this function is taken to be the accuracy obtained with a validation data which contains 58 samples in case of Hungarian data and 60 samples in case of Cleveland data (20% of the total data set size).

Now there are two steps to optimize this function first is to make the function return accuracy and make the particles move towards a maxima that is whenever the current global best or current personal best for the particle gets lower than value at the current position of the particle then update it and the second step is to make the function return negative of the accuracy and minimize it. Thus, optimizations are done to maximize the accuracy. The particle swarm optimization is used with 25 iterations and a swarm size of 40. The range of the dimension for the neural network layer and the RBM layer are taken to be in the range (6,12) and (15,60) respectively.The size of layers which are obtained are within this range only.

The flow chart of proposed srategy is given in figure 2.10.

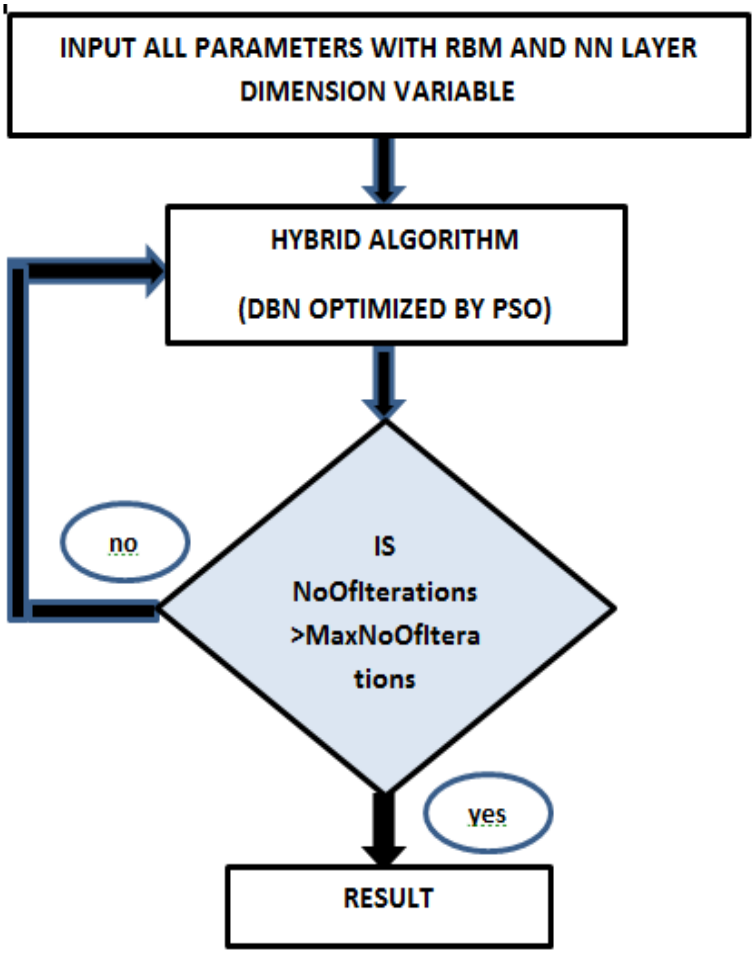


Figure 2.10: Proposed hybrid algorithm

2.5.1 Pseudo code

ModifiedTrainSupervisedDBN (Z, C, epsilon_CD, epsilon_C, L, n, W, b, V)

1. *let X the marginal over the input part of Z*
2. *PreTrainSupervisedDBN (X, epsilon_CD, L, n, W, b)*
3. *DBNSupervisedFineTuning (Z, C, epsilon_C, L, n, W, b, V, c)*
4. *Calculate accuracy of DBN with a testing data using W*
5. *return accuracy*

ModifiedTrainSupervisedDBNwithPSO(Z,C,epsilonCD,epsilonC,L,W,b,v,range_n, swarmsize,iter)

```
1.    let vector s denote the swarm position of the particles
2.    vector v denotes the velocity vector of the particles
3.    initialize global_best = -infinity
4.    initialize personal_best = -infinity
5.    for i<swarmsize
6.        initialize s[i] in range_n
7.        increment i
8.    while (iterations < iter)
9.        for i<swarmsize
10.         fitness_val=modifiedTrainSupervisedDBN(Z,C, epsilonCD, epsilonC,L, s[i],W,b,V)
11.         if fitness_val>particle_best
12.             set particle_best = fitness_val
13.          if particle_best > global_best
14.             set global_best=particle_best
15.         for i<swarmsize
16.             calculate the particle velocity for the particle i
17.             update s[i] according to v[i]
18.             use global best to update particle position
19.     return global_best that is the best dimension for the deep belief network
```

2.6 Implementation Details

1. Code is written in python in ubuntu.(Linux)
2. The python libraries used are pyswarm, numpy, scipy, sklearn.
3. Data is downloaded from UCI data repository (Hungarian and cleveland).

Chapter 3

RESULTS AND DISCUSSIONS

The designed hybrid algorithm is tested on two bench marked heart data sets of UCI repository [heart data set (uci)]-Cleveland data set and Hungarian data set .These two are carefully chosen as they are popular as well as they fit best for our classification algorithm. Our algorithm is designed using 80% of training data and 10% testing data. The proposed strategy proposed improves the accuracy.

3.1 Comparison of existing & proposed strategy

Differences in accuracy observed through introduction of nature inspired algorithm in DBN. In our proposed hybrid algorithm we have kept our dimensions of RBM and NN variable and then found the most optimal value of these using our modified DBN. This surely gave better results as it decreased the dependency on DBN to optimize these two important parameters. So time complexity of algorithm decreases. It is shown that in cleveland data set DBN alone gave accuracy of 88 % and with PSO it gave accuracy of 91.18 %.The dimensions which are obtained after optimization are 11 for RBM layer and 42 for NN layer. In hungarian data simple DBN gave an accuracy of 90.32 % and modified algorithm gave accuracy of 94.88%.The dimensions obtained after optimization by PSO are 6 for RBM layer and 27 for NN layer. We therefore realize that DBN can be further optimized by various nature inspired algorithm to improve accuracy. Figures (3.2 and 3.1) show results of our modified algorithm with hungarian and cleveland data set.

```
Best after iteration 15: [ 11.38351558  42.8382534 ] -0.874814814815
Best after iteration 16: [ 11.38351558  42.8382534 ] -0.874814814815
Best after iteration 17: [ 11.38351558  42.8382534 ] -0.874814814815
Best after iteration 18: [ 11.38351558  42.8382534 ] -0.874814814815
Best after iteration 19: [ 11.38351558  42.8382534 ] -0.874814814815
Best after iteration 20: [ 11.38351558  42.8382534 ] -0.874814814815
Best after iteration 21: [ 11.38351558  42.8382534 ] -0.874814814815
Best after iteration 22: [ 11.38351558  42.8382534 ] -0.874814814815
New best for swarm at iteration 23: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 23: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 24: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 25: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 26: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 27: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 28: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 29: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 30: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 31: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 32: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 33: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 34: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 35: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 36: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 37: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 38: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 39: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 40: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 41: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 42: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 43: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 44: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 45: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 46: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 47: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 48: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 49: [ 11.43235115  42.87225701] -0.911851851852
Best after iteration 50: [ 11.43235115  42.87225701] -0.911851851852
Stopping search: maximum iterations reached --> 50
(array([ 11.43235115,  42.87225701]), -0.9118518518518191)
-0.911851851852
```

Figure 3.1: Result of modified algorithm on cleveland data

```
Best after iteration 14: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 15: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 16: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 17: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 18: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 19: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 20: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 21: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 22: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 23: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 24: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 25: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 26: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 27: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 28: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 29: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 30: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 31: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 32: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 33: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 34: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 35: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 36: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 37: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 38: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 39: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 40: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 41: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 42: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 43: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 44: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 45: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 46: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 47: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 48: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 49: [  6.55047585  27.21032067] -0.948888888889
Best after iteration 50: [  6.55047585  27.21032067] -0.948888888889
Stopping search: maximum iterations reached --> 50
(array([  6.55047585,  27.21032067]), -0.94888888888888889)
-0.948888888889
```

Figure 3.2: Result of modified algorithm on hungarian data

3.2 Comparison of accuracies with previous work done on the data set

Our proposed method gave an accuracy of 91.18% for Cleveland data set and 94.88% for Hungarian data set.Earlier work in Hungarian data set showed maximum accuracy of 84 % using FDSS algorithm. Some other algorithms used in this data set are MLP-NN,FDSS,Ripper etc. . Similarly in cleveland data set maximum accuracy achieved is 87.50 % with ASVM . Some other algorithms used previously on this data set are : NN,C4.5,weighted fuzzy etc. Clearly or modified algorithm's results are better than previous work and can be used with other data sets like MNIST data set. The comparison among various existing algorithms and our hybrid algorithm on Cleveland data set is shown in table 3.1 and on hungarian data set in table 3.2. The pictorial representation of the results discussed above is shown in figure 3.3.

Table 3.1: **Comparison between existing and hybrid algorithm on cleveland dataset**

Algorithm	Accuracy %
ASVM [Mangasarian and Musicant (2000)]	87.24
SVMlight [Mangasarian and Musicant (2000)]	87.50
NN [Anooj (2012)]	53.86
FDSS[Setiawan et al. (2009)]	83
K-NN [Setiawan et al. (2009)]	81
C4.5 [Setiawan et al. (2009)]	82
Ripper[Setiawan et al. (2009)]	83
Weighted Fuzzy [Anooj (2012)]	57.85
Fuzzy [Anooj (2012)]	51.793
Deep belief network	**88**
DBN+PSO(proposed Stategy)	**91.18**

Table 3.2: **Comparison between existing and hybrid algorithm on cleveland dataset**

Algorithm	Accuracy
K-NN [Setiawan et al. (2009)]	92
FDSS[Setiawan et al. (2009)]	84
MLP-ANN [Setiawan et al. (2009)]	54
C4.5 [Setiawan et al. (2009)]	66
Ripper [Setiawan et al. (2009)]	68
Weighted Fuzzy [Anooj (2012)]	50.83
NN [Anooj (2012)]	46.417
Fuzzy [Anooj (2012)]	36
Deep belief network	**91.33**
DBN+PSO(proposed Stategy)	**94.88**

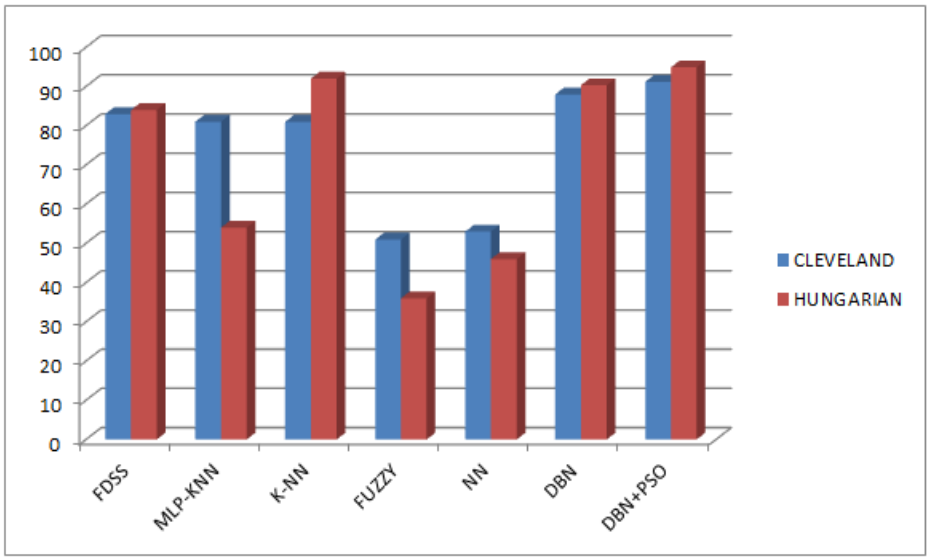


Figure 3.3: Comparison of various algorithms

3.3 Discussion about results

The algorithm used i.e. PSO for optimizing deep belief network is giving better results than simple deep belief network algorithm .This is because we have taken into account the optimization of variables in our hybrid algorithm .It uses swarm intelligence concept in its optimization to give accurate prediction.Our hybrid algorithm on UCI heart datasets gave better results than previously applied algorithms.Using nature inspired algorithms like PSO with deep learning and particularly deep belief network is a new way of looking into the problem of classification and diagnosis of diseases like heart diseases,brain tumours, cancer etc.

Chapter 4

CONCLUSION AND FUTURE WORK

4.1 Conclusion

In this project, Deep Belief network and its variants are studied to make a classifier for predicting heart diseases. Considering the limitations of neural network , the concept of deep learning is introduced. As a part of deep learning , deep belief networks are studied and analyzed to make a better performance classifier. We know that DBN is composed of RBM layers and NN layers and a large number of parameters are required to be adjusted for good performance . Optimizing these many parameters takes a lot of time and makes the performance worse. So in order to decrease the number of parameters to be optimized during DBN training ,particle swarm optimization is used to improve some of them . These parameters are the dimensions of RBM and NN layer to be used for unsupervised and supervised learning respectively.PSO can optimize multiple parameters at a time giving the most efficient results.

The proposed strategy of hybrid DBN algorithm with PSO successfully classified the heart disease type from the input data set provided.It predicted in a very reliable manner whether a person has heart disease or not reducing the dependency on traditional techniques .Machine learning algorithms like DBN can be used to make this diagnosis fast and less costly.This will help in detecting the symptoms of coronary disease before the patient goes for proper medical verification.This type of strategy will surely decrease the number of deaths due to heart diseases in developed and as well as developing countries.

4.2 Future Work

In this project, a more optimized version of Deep belief network is presented with PSO.This work can be extended by using other nature inspired algorithms like firefly , Ant colony Optimization(ACO),Flower pollination.

Also the system can be implemented practically.This will then take data in real time as well as predict or classify whether a person has a heart disease or not. Larger data sets can be managed using big data concepts .Also in real time data will be large so proper storage and processing of big data may become necessary.

Bibliography

[1] Anooj, P.: 2012, Clinical decision support system: risk level prediction of heart disease using decision tree fuzzy rules, *Int J Res Rev Comput Sci* **3**(3), 1659–1667.

[2] Bai, Q.: 2010, Analysis of particle swarm optimization algorithm, *Computer and information science* **3**(1), 180.

[3] Bengio, Y.: 2009, Learning deep architectures for ai, *Foundations and trends® in Machine Learning* **2**(1), 1–127.

[4] Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H. et al.: 2007, Greedy layer-wise training of deep networks, *Advances in neural information processing systems* **19**, 153.

[5] Bhatia, S., Prakash, P. and Pillai, G.: 2008, Svm based decision support system for heart disease classification with integer-coded genetic algorithm to select critical features, *Proceedings of the World Congress on Engineering and Computer Science, WCECS*, pp. 22–24.

[6] Carbonell, J. G., Michalski, R. S. and Mitchell, T. M.: 1983, An overview of machine learning, *Machine learning*, Springer, pp. 3–23.

[7] Eberhart, R. C., Kennedy, J. et al.: 1995, A new optimizer using particle swarm theory, *Proceedings of the sixth international symposium on micro machine and human science*, Vol. 1, New York, NY, pp. 39–43.

[8] Gardner, M. W. and Dorling, S.: 1998, Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences, *Atmospheric environment* **32**(14), 2627–2636.

[9] Hagan, M. T., Demuth, H. B., Beale, M. H. and De Jesús, O.: 1996, *Neural network design*, Vol. 20, PWS publishing company Boston.

[10] heart data set, U.: uci, Uci heart data set.
URL: *https://archive.ics.uci.edu/ml/datasets/Heart+Disease*

[11] Hinton, G.: 2011, Deep belief nets, *Encyclopedia of Machine Learning*, Springer, pp. 267–269.

[12] Hinton, G. E., Osindero, S. and Teh, Y.-W.: 2006, A fast learning algorithm for deep belief nets, *Neural computation* **18**(7), 1527–1554.

[13] Huang, G.-B. and Chen, L.: 2008, Enhanced random search based incremental extreme learning machine, *Neurocomputing* **71**(16), 3460–3468.

[14] Huang, G.-B., Zhu, Q.-Y. and Siew, C.-K.: 2006, Extreme learning machine: theory and applications, *Neurocomputing* **70**(1), 489–501.

[15] Jagtap, V. N. and Mishra, S. K.: 2014, Fast efficient artificial neural network for handwritten digit recognition, *International Journal of Computer Science and Information Technologies* **5**(2), 2302–2306.

[16] Kala, R., Vazirani, H., Shukla, A. and Tiwari, R.: 2010, Offline handwriting recognition using genetic algorithm, *arXiv preprint arXiv:1004.3257* .

[17] Kennedy, J.: 2011, Particle swarm optimization, *Encyclopedia of machine learning*, Springer, pp. 760–766.

[18] Kim, J. W.: n.d., Classification with deep belief networks.

[19] LeCun, Y., Bengio, Y. and Hinton, G.: 2015, Deep learning, *Nature* **521**(7553), 436–444.

[20] Lichman, M.: 2013, UCI machine learning repository.
URL: *http://archive.ics.uci.edu/ml*

[21] Maind, S. B. and Wankar, P.: 2014, Research paper on basic of artificial neural network, *International Journal on Recent and Innovation Trends in Computing and Communication* **2**(1), 96–100.

[22] Mangasarian, O. L. and Musicant, D. R.: 2000, Active support vector machine classification, *NIPS*, pp. 577–583.

[23] McCulloch, W. S. and Pitts, W.: 1943, A logical calculus of the ideas immanent in nervous activity, *The bulletin of mathematical biophysics* **5**(4), 115–133.

[24] Noor Akhmad, S., Venkatachalam, P. and Ahmad Fadzil, M. H.: 2009, Diagnosis of coronary artery disease using artificial intelligence based decision support system.

[25] Ribeiro, P. F. and Schlansker, W. K.: 2003, A hybrid particle swarm and neural network approach for reactive power control, *search. cpan. org/src/KYLESCH/AI-PSO-0.81/ReactivePower-PSO-wks. pdf* .

[26] Riley, L. and Cowan, M.: 2014, Noncommunicable diseases country profiles 2014, *Geneva: World Health Organization* .

[27] Rini, D. P., Shamsuddin, S. M. and Yuhaniz, S. S.: 2011, Particle swarm optimization: technique, system and challenges, *International Journal of Computer Applications* **14**(1), 19–26.

[28] Rosenblatt, F.: 1958, The perceptron: a probabilistic model for information storage and organization in the brain., *Psychological review* **65**(6), 386.

[29] Saeed, A.-M.: n.d., Intelligent handwritten digit recognition using artificial neural network.

[30] Sakshica, D. and Gupta, K.: n.d., Handwritten digit recognition using various neural network approaches.

[31] Schmidhuber, J.: 2015, Deep learning in neural networks: An overview, *Neural Networks* **61**, 85–117.

[32] Setiawan, N. A., Venkatachalam, P. and Hani, A. F. M.: 2009, Diagnosis of coronary artery disease using artificial intelligence based decision support system, *proceedings of the international conference on man-machine systems (ICoMMS), Batu Ferringhi, Penang*.

[33] Shi, X. H., Liang, Y. C., Lee, H. P., Lu, C. and Wang, Q.: 2007, Particle swarm optimization-based algorithms for tsp and generalized tsp, *Information Processing Letters* **103**(5), 169–176.

[34] Shukla, A., Tiwari, R. and Kala, R.: 2010, *Real life applications of soft computing*, CRC Press.

[35] Sivanandam, S. and Deepa, S.: 2007, *PRINCIPLES OF SOFT COMPUTING (With CD)*, John Wiley & Sons.

[36] Soni, J., Ansari, U., Sharma, D. and Soni, S.: 2011, Predictive data mining for medical diagnosis: An overview of heart disease prediction, *International Journal of Computer Applications* **17**(8), 43–48.

[37] Swietojanski, P., Ghoshal, A. and Renals, S.: 2014, Convolutional neural networks for distant speech recognition, *IEEE Signal Processing Letters* **21**(9), 1120–1124.

[38] Yusuf, S., Reddy, S., Ôunpuu, S. and Anand, S.: 2001, Global burden of cardiovascular diseases part i: general considerations, the epidemiologic transition, risk factors, and impact of urbanization, *Circulation* **104**(22), 2746–2753.

[39] Ziasabounchi, N. and Askerzade, I.: 2014, Anfis based classification model for heart disease prediction, *International Journal Of Electrical & Computer Sciences IJECS-IJENS* **14**(02), 7–12.