Chapter 5
Neural Network Based Expert Systems

Chapter Abstract
We already have stated in the chapter 3 of domain analysis that, for the domain of motivational strategies, neural network based expert has sufficient ability to solve the problem under consideration. Now, we will throw little more light on neural networks in this chapter as a building block of our expert system. The chapter discusses the concepts of artificial neural networks (ANN), features of neural networks and how neural network learns. The chapter also highlights different types of neural networks along with different algorithms for training them. The chapter mainly focuses on the back propagation (BP) algorithm, which we used for learning in solving our problem. The BP algorithm is designed for training multilayer neural networks and calculating weights for hidden and output layers. We also discussed different parameters of back propagation algorithm, which can be manipulated during implementation. The chapter also discusses the development process of neural network based expert system with respect to its lifecycle. The chapter also throws light on problems related to expert system development and critical success factors. The chapter also highlights how we manage risk and uncertainty in our project of neural network based expert system development. At the end, the chapter provides the prerequisites for successful implementation and version management of the expert system development project.

5.1 Difference between human and machine intelligence
We will soon explore the concept of neurons and Artificial Neural Network. As our primary aim is to generate machine intelligence which can replace human intelligence of HR managers, we need to clearly differentiate between the two. The table 5.1 provides the difference between human and machine intelligence. (Akerkar, 2007)
Table 5.1 Difference between human and machine intelligence

<table>
<thead>
<tr>
<th>Human Intelligence</th>
<th>Machine Intelligence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human being perceives everything as a pattern.</td>
<td>Machine perceives everything as a data.</td>
</tr>
<tr>
<td>Even for data, human tends to perceive a pattern.</td>
<td>Even for pattern, machine tends to perceive it as a data.</td>
</tr>
<tr>
<td>If there is no pattern, it is very difficult for a human being to remember and reproduce the data later.</td>
<td>The data is stored in machine’s memory, so the machine can reproduce the data at a point of time.</td>
</tr>
<tr>
<td>The pattern nature in storage and recall gives robustness and fault tolerance for the human system.</td>
<td>The data storing doesn’t give robustness and fault tolerance to machine.</td>
</tr>
<tr>
<td>Human being continuously learns from examples.</td>
<td>Continuous learning by examples is difficult to implement through algorithms on machines.</td>
</tr>
<tr>
<td>Human being can frame the patterns, even when the data are noisy or deformed due to the variations.</td>
<td>The machine cannot recognize any pattern from data when the data are noisy or deformed due to the variations.</td>
</tr>
</tbody>
</table>

5.2 Concept of ANN (Artificial neural network)

ANNs are a form of computation inspired by the structure and function of the brain. The human brain is densely interconnected network of approximately $10^{11}$ neurons; each connected to on an average $10^4$ others.

Artificial Neural Networks (ANN) are networks of interconnected simple units that are loosely modelled on greatly simplified idea of the brain’s functioning. ANNs are composed of units (nodes) and can be described as mapping an input space to an output space. (Keller, 2007)

5.2.1 The neurons

The human body is made up of a vast array of living cells. Certain cells are interconnected in a way that allows them to communicate pain, other tell the brain that they are experiencing cold, heat or other sensations. (Keller, 2007) These specialised communication cells are called neurons.

Neurons are connected to other neurons. In the terminology of system, the neuron can be thought of as a small computing engine that takes in inputs, process them and then transmits an output. The transfer function like sigmoid or logistic is used for transfer of output from one neuron. (Keller, 2007)
The other name for ANNs is data driven computational engines. Because ANNs were created to permit machines to form decision boundaries with their associated class regions as derived from the data. (Keller, 2007) The figure 5.1 presents simple neuron model with decision boundaries.

![Simple neuron model with decision boundaries](image)

**Figure 5.1** Simple neuron model with decision boundaries

### 5.3 Features of ANN

The following are the features of ANN.

- The trained ANN is able to construct an input - output mapping for the particular problem at hand.
- The ANN can work for input signals which are inherently non-linear.
- They have capability to adapt to the change in the environment.
- They learn by example.
- They constitute a distributed, associative memory.
- They are fault tolerant.
- They are capable of pattern recognition.
- They are well suited for VLSI technology.
• The contextual information is easily dealt with ANN, because a change in one neuron will affect the change in entire ANN.
• ANNs can generalise over input values. It means that in our neural network based expert system development case, they see the input value similar to other input values; they give same output.

5.4 Advantages of ANN

ANNs is the most effective to be used when solving new kind of problem that are difficult to simulate using logical and analytical techniques and whose solution is very difficult to define. The following are the advantages of using ANN.

• ANNs learn to recognise the patterns in the data set. They are developed through learning rather than programing.
• ANNs consumes less time of programmer or analyst as they can teach themselves.
• ANNs are flexible in changing environment. It may take some time to learn a sudden drastic change, but they can adjust themselves into constantly changing information.
• ANNs can build informative models where conventional approaches fail.

5.5 Limitations of ANN

The most common limitation of using ANNs is inability to provide explanation and inability to provide and explicit model for the problem domain. The following are the other limitations of using ANN.

• ANNs have inability to explain the model.
• It is difficult to extract rules from ANNs.
• Data used to train ANNs should be appropriate and measured in a way that reflects the behaviour of the factors. If data are not representative of the problem, neural computing will not produce accurate results.
• The time taken to train ANN model is high for a complex data set.
5.6 Application areas of ANN

The following are the few application areas of ANN.

- Tax form processing to identify the tax fraud.
- Enhancing auditing by finding irregularities.
- Bankruptcy prediction.
- Customer credit scoring
- Loan approval.
- Credit card approval and fraud detection.
- Financial prediction
- Energy forecasting.
- Intrusion detection and classification of attacks.
- Fraud detection in mobile telecommunication networks
- Performance appraisal of human resources
- Evaluating motivational strategies from employees’ perspectives.

5.7 How neural network learns?

ANNs are composed of interconnected units, which serve as model neurons. Each unit converts the pattern of incoming activities that it receives into a single outgoing activity that it broadcast to other units. It performs this conversation in two stages. (Akerkar, 2007)

i. It multiplies each incoming activity by the weight on the connection and adds together all these weighted inputs to get a quantity called the total input.

ii. A unit uses an input-output function that transforms the total input into the outgoing activity.

The behaviour of an ANN depends on both the weights and the input-output function (transfer function) that is specified in the units. This function typically falls into one of three categories.

- Linear
- Threshold
- Sigmoid
**Linear:** - For linear units, the output activity is proportional to the total weighted output.

**Threshold:** - The output is set at one of two levels, depending on whether the total input is greater than or less than some threshold value.

**Sigmoid:** - The output varies continuously, but not linearly as the input changes. The sigmoid function is the most common form of activation function used in ANN because of the following reasons. (Rao, 2011)

1. The smoothness of the function makes it easy to devise learning algorithms.
2. Observations show that biological neurons demonstrate firing rate, which is roughly sigmoidal.
3. From hardware and software implementation point of view, exponential functions are expensive and extensively computationally.

The commonest type of artificial neural network consists of three groups or layers of units. They are, a layer of input units is connected to a layer of hidden units, which is connected to a layer of output units. (Akerkar, 2007)

- The activity of the input units represents the raw information that is fed into the network.
- The activity of each hidden unit is determined by the activities of the input units and the weights on the connection between the input and the hidden units.
- The behaviour of the output units depends on the activity of the hidden units and the weights between the hidden and output units.

We can teach a three layer network to perform a particular task by using the following procedure. (Akerkar, 2007)

- We present the network with training examples, which consists of a pattern of activities for the input units together with the desired pattern of activities for the output units.
- We determine how closely the actual output of the network matches the desired output.
• We change the weight of each connection so that the network produces a better approximation of the desired output.

Now we will discuss the feed-forward network structure of ANN, which is considered one of the most popular ANN structure.

5.8 Feed forward neural networks

The figure 5.2 presents a multilayer feed forward neural network. (Keller, 2007) On the left portion of the figure are inputs to the first layer of neurons, followed by interconnected layers of neurons and finally with outputs from the final layer of neurons. Each layer directly supplies the next layer in the network, feeding the inputs forward through the network. Here neurons on the first layer could feed a neuron on the third as well as the second layer.

![Multilayer feed forward neural networks](image-url)

**Figure 5.2**  Multilayer feed forward neural networks
Typically a multilayer feed forward network consists of an input layer of source neurons, at least one middle or hidden layer of computational neurons and an output layer of computational neurons.

The input layer accepts input signals from outside world and redistributes these signals to all neurons in the hidden layer. Input layer rarely includes computing neurons and thus doesn’t process input patterns. The output layer accepts output signals, or stimulus pattern, from the hidden layer and establishes the output pattern of the entire network. (Negnevitsky, 2008)

Neurons In the hidden layer detect the features; the weights of the neurons represent the features hidden in the input pattern. These features are then used by the output layer in determining the output pattern.

With one hidden layer, we can represent any continuous function of the input signals, and with two hidden layers even discontinuous functions can be represented.

Feed forward networks feed output from individual neurons forward to one or more neurons or layers in the network. Networks that feed outputs from neuron back to the inputs of previous layers themselves or other neurons on their own layer are called recurrent networks.

The process of modifying the weights in a neuron or network to correctly perform a desired input-to-output mapping is termed learning in the neural network. (Keller, 2007)

5.9 Learning methods

There are two learning methods for neural network. They are: supervised training methods and unsupervised training methods. Apart from that there is one more classification called online and offline learning.
5.9.1 Supervised training

The process of using desired outputs for training the neural network is known as supervised training. Supervised training employs a teacher to assist in training the network by telling the network what the desired response to a given stimulus should be. The figure 5.3 shows the block diagram of supervised learning model. (Keller, 2007)

![Block diagram of supervised learning model](image)

**Figure 5.3** Block diagram of supervised learning model

The important principle is that supervised learning requires an input and a corresponding desired output. But in many situations, it is neither practical nor possible to train the learning system by using supervised training methods. In such cases, unsupervised learning methods are used.
5.9.2 Unsupervised training

The figure 5.4 shows the block diagram of unsupervised learning model. Here no teacher is employed in the training process. It is analogous to students learning the lessons on their own. The unsupervised method doesn’t need a desired output for each input factor. The adaptation rule in the unsupervised training algorithm performs the error-signal generation role; the teacher performs in the supervised learning system. (Keller, 2007)

![Block diagram of unsupervised learning model](image)

**Figure 5.4** Block diagram of unsupervised learning model

5.9.3 Offline or online training

In the offline learning, all the given patterns are used together to determine the weights. On the other hand, in an online learning the information in each new pattern is incorporated into the network by incrementally adjusting the weights. Thus an online learning allows the neural network to update the information continuously. But offline learning provides one advantage, i.e. it allows to extract information using all the training samples.
5.10 Back propagation neural network

The BPN model was first introduced by Elman, which is a two-layered back propagation network, with the addition of feedback connection from the output of the hidden layer to its input. (Padhy, 2005) The feedback path allows a BPN network to learn to recognise and generate temporal as well as spatial patterns. Since there is a feedback connection from the first layer output to the first layer input, a recurrent connection is established.

Back propagation training algorithm

Back propagation is a systematic method for training multiple layer artificial neural networks. The steps of standard back propagation algorithm are as follows. (Padhy, 2005)

1. Build a network with the chosen number of input, hidden and output units.
2. Initialise all the weights to low random values.
3. Choose a single training pair at random.
4. Copy the input pattern to the input layer.
5. Cycle the network so that the activation from the inputs generate the activations in the hidden and output layers.
6. Calculate the error derivative between the output activation and the target output.
7. Back propagate the summed products of the weights and errors in the output layer in order to calculate the error in the hidden units.
8. Update the weights attached to each unit according to the error in that unit, the output from the unit below it, and the learning parameters, until the error is sufficiently low or the network settles.

Back propagation training for a multilayer neural network: -

The training procedure is as follows. (Padhy, 2005)

1. Generate small random values (Both positive and negative) for the weights to ensure that the network is not saturated by large values of weights.
2. Choose a training pair from the training set.
3. Apply the input vector to the network input.
4. Calculate the network output
5. Calculate the error, i.e., the difference between the network output and the desired output.

6. Adjust the weights of the network in a way that minimizes the error.

7. Repeat steps 2 to 6 for each input-output pair in the training set until the error for the entire system is acceptably low.

The training of an ANN involves two passes. The first one is forward pass and the second one is backward pass.

**Forward pass:** - In the forward pass, the input signals move forward from the network input to the output.

**Backward pass:** - In the backward pass, the calculated error signals propagate backward through the network, where they are used to adjust the weights. (Swingler, 1996)

The calculation of the output is carried out, layer by layer, in the forward direction. The output of one layer is the input to the next layer as in feedback. In the backward pass, the weights of the output neurons layers are adjusted first, since the target value of each output neuron is available to guide the adjustment of the associated weights, using the delta rule.

Next, the weights of the middle layers are adjusted. But the middle layer neurons have no target value to adjust the weights. Hence, the training is more complicated, because the error must be back propagated through the network layer by layer using the non-linear function. The number of hidden units depends on the number of input units. The Kolomogorov’s theorem says that the number of hidden units as one greater than twice the number of input units. (Padhy, 2005)

### 5.10.1 Back propagation learning algorithm

The following steps present the back propagation learning algorithm. (S. Rajasekaran, 2012)
Step 1 Normalize the inputs and outputs with respect to their maximum values. It is proved that the neural networks work better if inputs and outputs lie between 0-1. For each training pair, assume there are ‘l’ inputs given by \( l \times 1 \{I\}_I \) and ‘n’ outputs \( n \times 1 \{O\}_O \) in a normalised form.

Step 2 Assume the number of neurons in the hidden layer to lie between \( 1 < m < 2l \).

Step 3 \([V]\) Represents the weights of synapses connecting input neurons and hidden neurons and \([W]\) represents weights of synapses connecting hidden neurons and output neurons. Initialise the weights to small random values usually from -1 to 1. For general problems, \( \lambda \) can be assume as 1 and the threshold values can be taken as zero

\[
[V]^0 = [\text{random weights}]
\]

\[
[W]^0 = [\text{random weights}]
\]

\[
[\Sigma V]^0 = [\Sigma W]^0 = [0]
\]

Step 4 For the training data, present one set of inputs and outputs. Presents the pattern to the input layer \( \{I\}_I \) as input to the input layer. By using linear activation function, the output of the input layer may be evaluated as

\[
\{O\}_I = \{I\}_I
\]

\[ L \times 1 \quad l \times 1 \]

Step 5 Computer the inputs to the hidden layer by multiplying corresponding weights of synapses as

\[
\{I\}_H = [V]^t \{O\}_I
\]

\[ m \times 1 \quad m \times l \quad l \times 1 \]

Step 6 Let the hidden layer units evaluate the output using the sigmoidal function as

\[
\{O\}_H = \left\{ \frac{1}{1+e^{-\lambda_{hi}}} \right\}
\]

Step 7 Compute the inputs to the output layer by multiplying corresponding weights of synapses as

\[
\{I\}_O = [W]^t \{O\}_H
\]

\[ n \times 1 \quad n \times m \quad m \times 1 \]

Step 8 Let the output layer units evaluate the output using sigmoidal function as

\[
\{O\}_O = \left\{ \frac{1}{1+e^{-\lambda_{lo}}} \right\}
\]

Step 9 Calculate the error and the difference between the network output and the desired output as for the \( i^{th} \) training set as
Design of expert system prototype for analysing and structuring motivational strategies on ICT human resources to reduce employee turnover ratio

\[ E_P = \sqrt{\frac{\varepsilon(T_j - O_{ij})^2}{n}} \]

**Step 10**
Find \{d\} as

\[ \{d\} = \{T_k - O_{ok}\}O_{ok}(1 - O_{ok}) \]

**Step 11**
Find \([Y]\) matrix as

\[ [Y] = \{O\}_{i\in<d>} \]
\[ \begin{array}{ccc}
m*n & m*1 & 1*n \\
\end{array} \]

**Step 12**
Find \([\Delta W]\)^{t+1} = \alpha[\Delta W]^t + \eta[Y]

**Step 13**
Find \{e\} = [W] \{d\}

\[ \{d^*\} = \{e_i(O_{Hi})(1 - O_{Hi})\} \]

**Step 14**
Find \([X]\) matrix as

\[ [X] = \{O\}_{i\in<d^*>} = \{I\}_{i\in<d^*>} \]
\[ \begin{array}{ccc}
1*m & 1*1 & 1*m \\
1*m & 1*m & 1*m \\
\end{array} \]

**Step 15**
Find

\[ [V]^{t+1} = [V]^t + [\Delta V]^{t+1} \]
\[ [W]^{t+1} = [W]^t + [\Delta W]^{t+1} \]

**Step 16**
Find error rate as

\[ error\ rate = \frac{\varepsilon E_P}{n_{set}} \]

**Step 17**
Repeat steps 4-16 until the convergence in the error rate is less than the tolerance value.
5.11 Parameters in back propagation algorithm

Here, we will discuss some of the important parameters of back propagation algorithm, which can affect the process of learning and performance of learning.

5.11.1 Rate of learning

The rate of learning is denoted by $\eta$, has direct bearing on the change in synaptic weights. The back propagation algorithm is an approximation to the trajectory in weight space. If we choose a small learning rate parameter, the trajectory will be smoother in the weight space, but the rate of learning will be very slow. On the other hand a large learning rate parameter will result in large changes in the synaptic weights and the network may show oscillatory response and become unstable. (Rao, 2011)

The selection of learning rate depends on the number and type of input patterns. An empirical formula to select learning coefficient is as follows: (H.A.C Eaton, 1992)

$$\eta = \frac{1.5}{N_1^2 + (N_1^2 + \cdots + (N_m^2)}$$

Where $N_1$ is the number of patterns of type 1 and $m$ is the number of different pattern types:

The learning rate must be smaller where there are many input patterns as compared to when they are few because the step length is controlled by the learning coefficient. The optimum value of learning rate is 0.6. (S. Rajasekaran, 2012)

5.11.2 Momentum rate

Momentum rate is denoted by $\alpha$, is used to improve the generalized delta rule. The inclusion of the momentum constant stabilizes the weights which tend to oscillate. The momentum rate is obtained by adding a fraction of the previous weight change to the current weight change.
The momentum rate has a significant role in deciding the values of learning rate that will produce rapid learning. It determines the step size of change in weights or biases. If momentum rate is zero, the smoothening is minimum and the entire weight adjustment comes from the newly calculate change.

If momentum rate is one, new adjustment is ignored and the previous one is repeated. Between zero and one is a region here the weight adjustment is smoothened by an amount proportional to the momentum rate. Momentum rate of 0.9 has been found to be suitable for most of the problems. (S. Rajasekaran, 2012)

The role of momentum rate is to increase the speed of learning without leading to oscillation. The momentum term effectively filters out high frequency variations of the error surface in the weight space, since it adds the effect of past weight changes on the current direction of movement in the weight space.

5.11.3 Threshold value
Threshold value of a neuron is denoting by \( \theta \), also called bias or the noise factor. A neuron fires or generates an output if the weighted sum of input exceeds the threshold value. One method is to simply assign a small value to it and not to change it during training. The other method is to initially choose some random values and change them during training. (S. Rajasekaran, 2012)

5.11.4 Mode of training
The training pattern can be described in two ways.
   i. Sequential mode: - Here the weight updating is performed on the presentation of each input pattern.
   ii. Batch mode: - In this mode, the weight updating is performed after presentation of all the input patterns.

5.11.5 Termination criteria
A termination criterion commonly used is to terminate when the absolute rate of change in the average squared error per epoch is sufficiently small.
5.11.6 Generalisation

A back propagation network is said to generalise when the input-output mapping computed by the network, is correct for test data not used in the training.

When a neural network learns too many input-output samples, the network may end up memorizing the training set, without trying to find the underlying function to be modelled. This is called as over fitting or over training. When over trained, the network loses the ability to generalize.

Generalization is influenced by

- Size of training set and how close it is to the environment of interest.
- Architecture of the neural network.
- Physical complexity of the problem

For good generalization, the size of training set $N$ is given by,

$$N = O\left(\frac{W}{\epsilon}\right)$$

Here $W$ is the total number of free parameters (synaptic weights and biases) in the network and $\epsilon$ is the error permitted on the test data, and $O(.)$ is the order of quantity enclosed within. (Rao, 2011)

5.11.7 Problem of local minima

The problem of local minima occurs because the algorithm always changes the weights in such a way as to cause error to fall. But the error might briefly have to rise as a part of more generic fall. In such a case, algorithm will get stuck and the error will not decrease further. The above phenomenon is called the problem of local minima. The problem can be solved by resetting the weights to different random numbers and then train the network again or introduce the momentum rate in the algorithm.
5.11.8 Cross validation
The Training data set has two parts. One of them is randomly selected records which are used for training the neural network. The other part is validation data set. The validation data set is used to avoid the problem of over training. When the trained weights for the training data set reaches a higher error over the stored weights of validation data set, then training is terminated and the stored weights are returned.

5.11.8 Number of hidden layers
The universal approximation theorem states that a single hidden layer is sufficient for a multilayer perceptron to compute a uniform function approximation to a given training set represented by the set of inputs and target output. However, a single hidden layer need not be optimum in terms of learning time, network implementation of generalization. (Rao, 2011)

A better back propagation network can be constructed when there are two hidden layers; where the first hidden layer will extract the local features and the second hidden layer will extract the global features.

5.12 Suitability of back propagation algorithm
Back propagation algorithm is most suited for problems with the following characteristics.
- Input is high dimensional real values.
- Output is real value.
- Output is a vector of values.
- There exists possible noisy data.
- Long training time is acceptable for the system.
- It is not very important for humans to understand the weights.

5.13 Advantages of back propagation algorithm
The following are the advantages of back propagation algorithm for training feed forward networks.
- It is easy to use, with few parameters to adjust.
- The algorithm is easy to implement.
- It is applicable to a wide range of problems.
- It is able to form arbitrarily complex non-linear mappings. (i.e., it is a universal approximator)
- It is popular and widely used for training feed forward networks as well as some recurrent networks.
- It performs stochastic gradient descent in weight space, pattern by pattern, by updating synaptic weights.
- It relies on local computations to discover the information processing capabilities of neural network.
- The hidden neurons of a multilayer perception act as a feature detector.
- It works well for functional approximation.
- It is robust, which means that disturbance with small energy can only give rise to small estimation errors.

5.14 Disadvantages of back propagation algorithm

The following are the disadvantages of back propagation algorithm for training feed forward networks.

- There is an inability to know how to precisely generate any arbitrary mapping procedure.
- It is hard to know how many neurons and layers are necessary.
- Learning can be slow.
- New learning will overwrite old learning unless old patterns are repeated in the training process.
- It has no inherent novelty detections so it must be trained on known outcomes.

5.15 Knowledge engineering for neural network based expert system

The process of building intelligent knowledge base system is called knowledge engineering. The approach for developing the conventional system is known as SDLC (System development life cycle). While the approach for developing an expert system is known as knowledge engineering approach. The knowledge
Design of expert system prototype for analysing and structuring motivational strategies on ICT human resources to reduce employee turnover ratio

Viral Nagori

Computer Science
Registration Number – 09B0150

5.15.1 Problem assessment

The phase has four main activities. (Negnevitsky, 2008)

- Determine the problem’s characteristics
- Identify the main participant in the project
- Specify the project objectives
- Determine the resources needed for building the system.

Determine the problem characteristics:

The first step in knowledge engineering phase is to determine the problem type before determining the problem characteristics. Typically problem types are: diagnosis, selection, prediction, classification, clustering, optimisation and control. Our neural network based expert system will be classified under the problem type of prediction and classification. As our neural network based expert system predicts whether an employee will prefer a particular motivation strategy or not and then classifies him/her into motivated or non-motivated employee.

Identify the main participants in the project:

The main participants in our neural network based expert systems are HR managers, ICT employees, me and my research guide. HR managers are the domain expert, who provided us with insight and listing of motivation strategies employed in the ICT organisations. ICT employees have provided their preferences on motivational strategies, which formed the basis for our training, testing and validation data set. We, i.e. me and my research guide acted as knowledge engineers. We implemented engineering is a six phase process. (Durking, 1994) Here we are describing the major activities in each phase and the mapped them to activities we carried out in development of neural network based expert system.

1. Problem assessment
2. Data and knowledge acquisition
3. Development of a prototype system
4. Development of a complete system
5. Evaluation and revision of the system
6. Integration and maintenance of the system
back propagation algorithm of neural network and deciding various parameters like learning rate, momentum rate, number of hidden layers, number of nodes in hidden layers and activation functions.

Specify the project objectives:
The main objective of our project is to provide automated solution which can evaluate motivational strategies from employees’ perspectives and in turn helps HR managers to structure their motivational strategies in such a way to increase employee retention and reduce employee turnover ratio.

Determining the resources needed for building the system:
The resources, which we required were laptop with 4 GB RAM and high processing speed, MATLAB as development software, literatures on Neural network based expert systems, textbooks and web materials to understand and implement neural network, web hosting and server space to host our website for data collection and money to buy the above resources.

5.15.2 Data and knowledge acquisition
The phase has two main activities. (Negnevitsky, 2008)
- Collect and analyse data and knowledge
- Make key concepts of the system design more explicit

Collect and analyse data and knowledge:
The data, which we have to collect, are from primary sources. The primary sources of data are ICT employees working in different ICT companies. While collecting the data, we converted them into nominal scales before storing it in the database. We wanted to data on employees’ demographic factors and their preference for each motivational strategy. We wanted information like age, gender, education qualification, marital status, current level of hierarchy, and current salary. For example, we converted gender into nominal scale. Let say 1 for male and 2 for female. Similarly, for age, we used the following nominal scales: 1 for age from 18 to 25, 2 for age from 26 to 35, 3 for age from 36 to 50, and 4 for age above 50 years. The conversion of demographic factors into nominal scale is given in the chapter 4
of research methodology. Similarly, we are storing the preferences for motivational strategies in yes or no. We are converting yes into 1 and no into 0.

During the data collection, if we encounter the inconsistent and missing data, we have to discard the entire record of that particular employee. Let say if employee has not provided his current salary figure, then we can’t just assume an average value and convert into a nominal scale and then store it in our database. The wrong values and wrong record in our training set might lead to wrong training for neural network and in turn will result into wrong prediction of the output.

**Make key concepts of system design more explicit:**

The key knowledge areas of our neural network based expert system are listing of motivational strategies and classifying them, and employees’ demographic factors. The key concepts related to neural network system design are inference engine, which is the neural network component itself and knowledge base, which consists of information about employees’ demographic factors and employees’ preferences on motivational strategies.

**5.15.3 Development of a prototype system**

This phase involves creating an intelligent system or rather a small version of it – and testing it with a number of test cases. The major activities in this phase are: (Negnevitsky, 2008)

- Choose a tool for building an intelligent system
- Transform data and represent knowledge
- Design and implement a prototype system
- Test the prototype with test cases.

**Choose a tool for building an intelligent system:**

The tool, which we are going to use to develop neural network based expert system, is C++ and MATLAB.
Transform data and represent knowledge:
The data on employees’ demographic factors and their preferences are placed into excel sheet. Each excel sheet belongs to each individual motivational strategy, where information about the related demographic factors for that motivational strategy and employees’ preferences for that motivational strategy are stored. Retrieving the stored data from database to excel sheet is also easier.

Design and implement a prototype system:
A prototype is a small working version of the final system. It is designed to test how well we understood the problem, and our choice of problem solving strategy, the tool selected for building a system, and techniques for representing acquired data and knowledge are adequate or not.

We initially developed a prototype on a very small scale for our neural network based expert system. We called this prototype as concept proofing our approach to solve the problem at hand. The prototype was tested successfully, but due to inherent limitations of C++ to handle large data and matrix multiplication is very time consuming, which increases the learning time of the system, we implemented our prototype using MATLAB and tested it for different test cases. At the prototype stage, we just developed our neural network based expert system for two motivational strategies out of 47 motivational strategies.

Test the prototype with test cases:
We tested our initial prototype with 20 test cases, whose results are known to us. In our test cases, we have information about employees’ demographic factors and their preferences on motivational strategies. The results were tested for accuracy and consistency and found to be satisfactory in line with the results of other such systems.

5.15.4 Development of a complete system
Once the results of the prototype are functionally satisfactory, the next step is to develop a plan, schedule and budget for the complete system and define the system’s performance criteria.
The major activities in this phase are as follows. (Negnevitsky, 2008)

- Prepare a detailed design for a full scale system.
- Collect additional data and knowledge.
- Develop the user interface.
- Implement the complete system.

**Prepare a detailed design for a full scale system:**
We developed separate excel files for each motivational strategy having details about employees’ demographic information and their preferences on motivational strategy. We identified and decided the values of important parameters like momentum rate, learning rate, activation function, number of nodes in hidden layer and number of hidden layers for our back propagation neural network.

**Collect additional data and knowledge:**
We collected data of around more than 2000 ICT employees on their demographic factors and their preferences on motivational strategies. We also collected data on HR managers’ perspectives on which combinations of demographic factors will act as inputs for which motivation strategy.

**Develop the user interface:**
Our user interface is command line having text based inputs and instructions. The system is going to be used by HR managers for knowing preferences of employees on motivational strategies, it will be used internally by the department. Hence system does not require graphical user interface.

**Implement the complete system:**
We have implemented the complete system in MATLAB and the system is running perfectly with the desired output is achieved in terms of accuracy and consistency.

**5.15.5 Evaluation and revision of the system**
The major activities in this phase are: (Negnevitsky, 2008)

- Evaluate the system against the performance criteria.
- Revise the system as necessary
Evaluate the system against the performance criteria:
We evaluated our system against the performance criteria. The performance criteria for neural network based expert system is set to be 80% or above in terms of accuracy. Our system meets these criteria.

Revise the system as necessary:
We revised the system later on to add the functionalities like knowledge update facility, explanation facility and batch processing for learning of system for 47 motivational strategies.

5.15.6 Integration and maintenance of the system
This is the last phase in development of the system. It involves integrating the system into the environment where it will operate and establishing an effective maintenance program.

The phase has following main activities. (Negnevitsky, 2008)

- Make arrangement of technology transfer
- Establish an effective maintenance program

Make arrangement of technology transfer:
By integrating means, we need to provide interface for new intelligent system with existing systems in the organisation. In our case there is no existing system which can automate the task of evaluation of motivational strategies from employees’ perspectives.

Establish an effective maintenance program:
As knowledge base system evolves over time because of change in knowledge. In our system, we have provided knowledge update facility and facility of relearning in the batch mode. Hence, we have taken effective measures for maintenance of our system.

The figure 5.5 shows the phases of knowledge engineering and mapping of its activities in the development of our neural network based expert system.
<table>
<thead>
<tr>
<th>Phases</th>
<th>Activities</th>
<th>Mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem Assessment</td>
<td>Determine the problem characteristics</td>
<td>Problem type is prediction and classification.</td>
</tr>
<tr>
<td></td>
<td>Identify the main participant in the project</td>
<td>HR managers as domain experts, ICT employees, me and my research guide as knowledge engineers</td>
</tr>
<tr>
<td></td>
<td>Specify the project objectives</td>
<td>Increase employee retention and reduce employee turnover ratio</td>
</tr>
<tr>
<td></td>
<td>Determine the resources needed for building the system</td>
<td>Hardware, software, web materials, textbooks, web server and money</td>
</tr>
<tr>
<td>Data and knowledge acquisition</td>
<td>Collect and analyze data and knowledge</td>
<td>Employees' demographic factors and their preferences on motivational strategies</td>
</tr>
<tr>
<td></td>
<td>Make key concepts of system design more explicit</td>
<td>Inference engine and knowledge base</td>
</tr>
<tr>
<td>Development of a prototype system</td>
<td>Choose a tool for building an intelligent system</td>
<td>C++ and MATLAB</td>
</tr>
<tr>
<td></td>
<td>Transfer data and represent knowledge</td>
<td>Prepare excel sheets</td>
</tr>
<tr>
<td></td>
<td>Design and implement a prototype system</td>
<td>Prototype with 2 motivational strategies</td>
</tr>
<tr>
<td></td>
<td>Test the prototype with test cases</td>
<td>20 test cases and tested for accuracy and consistency</td>
</tr>
<tr>
<td>Development of a complete system</td>
<td>Prepare a detailed design for a full scale system</td>
<td>Define parameters for back propagation network</td>
</tr>
<tr>
<td></td>
<td>Collect additional data and knowledge</td>
<td>More than 2000 records in the training data set of neural network</td>
</tr>
<tr>
<td></td>
<td>Develop the user interface</td>
<td>Command line text base user interface</td>
</tr>
<tr>
<td></td>
<td>Implement the complete system</td>
<td>MATLAB</td>
</tr>
<tr>
<td>Evaluate and revision of the system</td>
<td>Evaluate the system against the performance criteria</td>
<td>80% above accuracy and consistency</td>
</tr>
<tr>
<td></td>
<td>Revise the system as necessary</td>
<td>Knowledge update facility, explanation facility and batch processing</td>
</tr>
<tr>
<td>Integration and maintenance of the system</td>
<td>Make arrengement of technology transfer</td>
<td>No old existing system</td>
</tr>
<tr>
<td></td>
<td>Establishing an effective maintenance program</td>
<td>Knowledge evolution and updation</td>
</tr>
</tbody>
</table>
5.16 Project management for neural network based expert system

Development of systems, whether it is a conventional system, an embedded system or intelligent systems like expert systems requires project management. In the most generic terms, project management is about successful execution and implementation of the project. (Daniel Bobrow, 1986) As far as software development is concerned, project management is about planning, identifying requirements, development, testing and implementation, and evaluation of the system. (Vikas Arora, 1991)

The above definition of project management is almost similar to the system development life cycle (SDLC) of software development. So before discussing the issues related to project management, we will first introduce the concept of SDLC.

5.16.1 System development life cycle

Though the word system development life cycle is used to for structured development of the system and is the most preferred approach when requirements are clearly defined and the development proceeds in the linear way; where the next phase is executed after the completion of the previous phase. (N. Ashrafi, 1995)

The second approach is object oriented approach which helps in modular development of the system. The object oriented approach, which is most preferred when requirements are not very clearly defined, or user can put requirements in to verbatim. The second reason to use this approach is that when there are high chances that developed system will undergo rapid changes due to change in requirements or change in environments. Most of the expert system requires frequent updates in knowledge base as knowledge evolves over a period of time.

When an expert system is used in the real time environment, then it is preferred to model the object using object oriented approach. (Eddie Washington, 1989) Object oriented approach also works when the problem at hand is not properly defined and the experts are not clear about the solution and performance criteria of the system.

Here we are not seeing structured and object oriented as an approach, but methodology, which can be combined for the development of the system. The both
above methodology can be applied to any system development project, whether large, and complex or small and simple. (Garrett, 1991) Hence we are defining neural network based expert system development life cycle for our research.

The conventional SDLC has five phases. They are:

1. System planning
2. System analysis
3. System design
4. System development
5. System testing
6. System implementation
7. System evaluation and maintenance

Based on the same line, we will briefly introduce the neural network based expert system development life cycle. We term it as NNES DLC.

5.17 NNES DLC

The phases of NNES DLC are:

1. Problem identification and justification
2. Proof of concept
3. Prototype development
4. Enhance the prototype
5. Fine tuning of the prototype
6. Convert prototype into full system
7. System evaluation and evolution

The explanation of each of this phase is given here by based on the neural network based expert system we developed for evaluating and structuring motivational strategies on ICT human resources.

5.17.1 Problem identification and justification

Before selecting the problem domain and justifying the need for neural network based expert system, we required to answer the following questions.
Design of expert system prototype for analysing and structuring motivational strategies on ICT human resources to reduce employee turnover ratio

Viral Nagori

Computer Science
Registration Number – 0980150

- Does the problem at hand really require the use of expert knowledge?
- Is the required expertise is very scarce and is it very hard to find the real expert of the domain?
- Do you think conventional computerised system will solve the problem?
- Does the solution to the problem is very critical for the organisation?
- Does the failure on the part of system is acceptable sometimes?
- Does it require to distribute expertise at many locations?
- Does knowledge required to develop an expert system change from time to time?
- Do you think the problem to be addressed pertaining to a very narrow domain?
- Does the expert system offer tangible and intangible benefits which exceed the cost?

Now, we are going to answer the above question for the research problem we addressed. For our research, we are working in human resource domain, and the policies and strategies keep changing over the period of time. Not only has that, employees’ perspectives and HR managers’ perspectives also changed from time to time and situation to situation. (Nagori, 2012) Hence it is very difficult to define rules.

Secondly, an expert system should work with a small and narrow domain. (Rolston, 1988) Hence we are restricting our problem domain to motivational strategies of human resource domain. The problem in this domain is that despite the best of the motivational strategies, employee turnover ratio is very high. (Viral Nagori, 2012) While interacting with the ICT employees, we found that the problem is that motivational strategies are designed from employers and HR manager’s perspectives and perspectives of employees and their valuable inputs are not considered. Hence the motivational strategies implemented failed to motivate and retain the employees.

So, one of the solution is to consider employees’ perspectives before formulating motivational strategies. Now let us consider a hypothetical, but very much possible
Design of expert system prototype for analysing and structuring motivational strategies on ICT human resources to reduce employee turnover ratio

scenario in the company. Let say, on an average, there are 200 employees working in the company, and company implements 20 motivational strategies to motivate employees. Even if HR manager interacts with four employees in a day and spend around one hour in communication with each employee, then he will require at least two months in just interacting and knowing the perspectives of an employee. Meanwhile, based on the employees’ perspectives, if he/she formulates motivational strategies and implement, it might be very much possible that the perspectives would have changed and he has to restart the process. (Viral Nagori, Novice approach for evaluating motivational strategies from employees’ perspectives based on their demographic factors using back propagation algorithm, 2013)

Hence, there is a need for automating the solution, which can evaluate motivational strategies from employees’ perspectives. Thus we justified the need for neural network based expert system for evaluating motivational strategies from employees’ perspectives on ICT human resources.

The need to use neural network based approach is that the rules formulation is very difficult in answering whether an employee will be motivated by a particular motivation strategy or not. (Viral Nagori, Which type of expert system- rule based, fuzzy or neural is most suited for evaluating motivational strategies on human resources: - an analytical case study, 2012)

5.17.2 Proof of concept

In this phase, we need to select the tool and technology to demonstrate whether the choice of tool and technology is correct to address the problem. Company might start the expert system development project on the large scale and then later on it might happen that the efforts in terms of time and cost are not producing any desired solution for the problem. (J.A. Fenn, 1991) So it is always advisable to go for proof of concept for the tools and technology to be used for the solution of the problem.

From the literature review, we decided that for our neural network based expert system, back propagation algorithm is the best choice. In the initial stage to check whether back propagation algorithm is right technology to address the solution of
our problem domain by use of neural network based expert system, we implemented back propagation algorithm for pattern recognition in C++. We implemented back propagation algorithm to identify patterns of different digits from 0 to 9. And it generated the results as per our satisfaction.

Afterwards, we implemented back propagation algorithm in C++ for our problem domain, motivational strategies. We just selected one motivational strategy and having only 10 records in the training data set of neural network. The algorithm learning was successful and we tested on few records, and the answers were accurate. Hence we are very sure that back propagation algorithm can be used to develop neural network based expert system for evaluating motivational strategies on ICT human resources.

5.17.3 Prototype development
Once, the proof of concept is provided, the next step is to prepare the prototype of the system to test in the real environment, but with limited functionalities.

After the proof of concept was provided, we decided to build a prototype on live data. The prototype can be developed in C++ also, but C++ has inherent problems of handling very large real values and the time required for matrix calculation is also longer. If we implement it in C++, then it might be possible that the learning time of our system will be longer, and because of a problem of containing large values in the variable, neural network system might not converge for a particular reason.

From the literature review, we found that to implement the solution of neural networks, MATLAB and Scilab are two tools. But the resource in the form of tutorial and video demos for MATLAB can easily be found out; we decided to implement our prototype with MATLAB. (Viral Nagori, Expert systems, programing languages, tools and shells, 2014)

The prototype has been developed for evaluating two motivational strategies, with 200 records in the training data set and 20 records in the test data set. The results are
verified in terms of accuracy and consistency, and we found it satisfactory in comparison with the results of other neural network based expert systems.

5.17.4 Enhance the prototype

The next step is the NNESDLC is to enhance the prototype on the larger scale for real working environment. The enhancement of prototype is also known as incremental prototyping, where you add more features and functionalities. Here, we should consider all the possible aspects of real life scenario in developing the prototype. The enhanced prototype should work with more data on the training data set and more complex real life situations.

For our neural network based expert system, we enhanced our prototype for 47 motivational strategies in four different categories. The training data set has more than 200 records and testing data set has 40 records. We make our system learn twice for all motivational strategies and tested for 40 records in terms of accuracy and consistency. Again the results were at par with the results of other neural network based expert system. So we can say that our prototype is working efficiently.

5.17.5 Fine tuning of the prototype

Once enhanced prototype is working, the next step is to make prototype effective. Here some additional parameters affecting the system performance can be added or removed to check the performance of the system. The aim of this phase is to make system perfect as much as possible.

For our neural network based expert system, we introduced different activation functions, which can be used in back propagation algorithm. We also introduced different values of momentum rate, and learning rate to measure the learning performance of the system in term of error, number of epochs and the learning time. We also manipulated with number of nodes in the hidden layer to see the effect of system in learning and testing.
At the same time, we also introduce certain additional functionalities like knowledge update facility, batch processing and explanation facility to neural network based expert system.

5.17.6 Convert prototype into full system
In this system, prototype is converted into full working version of the system. Here user interface and help manuals are prepared for users’ interaction with the system. Once, the full fledge system is ready, the next issue is implementation of the project. The following implementation issues need to be taken care for implementation of neural network based expert system. (Craig K. Tyran, 1993)

- Commitment to project
- Management support
- User participation
- User training
- Maintenance

In our neural network based expert system, we created a command line user interface with text messages to help users to interact with the system. The full-fledged system is implemented and is able to provide results in the real environment. As our project is a research project, we have not implemented it yet commercially. Hence, we are not required to handle the implementation issues at this point of time.

5.17.7 System evaluation and evolution
The expert system should be evaluated after a certain period of its performance based using stakeholder based subjective assessment technique. (David Conrath, 1991) It involves both the parameters: Social and technical. The social factors relate to people and organisation. While technical factors relates to technical aspects like errors and defects, ease of use, documentation, system performance etc., related to neural network based expert system.

Any large system, whether it is an expert system or any conventional system, it evolves throughout its life. (Rolston, 1988). The evolution is may be because of additional functionalities are needed or because of change in requirements or change
in the environment in which system operates. The evolution is of the following types.

i. Functional evolution.
ii. Knowledge evolution.
iii. Domain evolution.
iv. User evolution

As this is the last phase, we cannot comment on the system evaluation and evolution as we have just implemented our system and as of now, there is no change in domain requirements, or user requirements or functional requirements, or knowledge requirements. We are in talks with few ICT companies to use our system in their environment. Once it is used by ICT companies, we can evaluate system performance in a better way.

The figure 5.6 will represent the NNES DLC. The process shows that the phases of NNES DLC are tightly integrated with each other.
Now, we will discuss the issues related to project management of neural network based expert system.

**Figure 5.6  Process of NNESDLC**
5.18 Issues related to project management of neural network based expert system

The project management of an expert system is altogether different compared to conventional system development project. (Larry G. Hull, 1991) Even the project management for rule based expert system and neural network based expert system differs. The project management here has been discussed in line with our developed neural network based expert system. The major issues related to project management for neural network based expert systems are given below.

- Critical success factors in expert system development
- Risk in expert system projects
- Uncertainty management issues
- Version management

The brief explanation about each of the issue is given below.

5.18.1 Critical success factors in expert system development

The critical success factors in expert system development vary vastly for different domains. The major success factors are the type of applications, the importance of the system to business strategy, organizational culture, organizational support, tool for development, and financial resources. (David Millett, 1996) Similarly, factors contributing to failure include poor project control, insufficient resources for maintenance, and financial constraints.

For us, critical success factor in expert system development was the data collection for preparing a training data set and test data set of neural networks. As the information we were collecting were personal information of employees, which are of private and sensitive in nature, we are not very sure how many companies will support us in our research study. It took us almost 6 months to approach and convince medium and large size companies to participate in our research study. But later on slowly and steady our efforts bought the results and one after another, companies have shown their willingness to participate in our research study.
5.18.2 Risk in expert system projects
The most crucial phase in expert system project management is identifying the major risk factors associated with ES project development. The risk factors can be internal due to organisation and employees or due to external environment. The risk factors cannot be completely avoided or eliminated, but if properly planned, then impact of risk can be reduced. (Douglass Hillmer, 1991) The risk factors can be classified based on quantitative terms as high value risks or low value risks. The impact of high value risks is cascading and may lead to heavy financial losses.

The most common risk factors associated with expert systems are the resistance from the in house expert who may feel threatened by the expert system development and legal liabilities in case of malfunctioning of an expert system. But in our designed and implemented neural network based expert systems, both the above risks are of low value. As our expert system is used for internal process of structuring motivational strategies, the liability of outside claim is zero even if our expert system malfunctions. The in house expert can be convinced that with the use of our expert system, his/her time to understand the employees’ perspective on motivational strategies can be significantly reduced. The saved time can be utilised in a more productive way to other HR functions. When we approached HR managers for data collection, they welcomed idea and they feel that if their time can be utilised in a productive way, then there is no threat to their position from expert system.

5.18.3 Uncertainty management issues
Handling uncertainty is an important managerial aspect in expert system project development. Expert systems can be divided into two classes with respect to uncertainty handling. In one class, the knowledge representation and inferencing mechanism are deterministic and decision making and reasoning can proceed categorically without explicit representation and propagation of uncertainty. The other class consists of those applications in which uncertainty is an inherent quality of the knowledge or reasoning process as such requires special mechanism for representation and inferencing. (Avanzato, 1991)
Uncertainties can arise because of unreliable information, imprecision in language used for representation, inferencing with incomplete information and integrating knowledge from various expert sources. The uncertainties can be measured in two ways. One of them is to use probabilities; associate some numeric value with uncertainty and second one is that of certainty factor model.

As far as our neural network based expert system development project is concerned, it is classified in the first category. In our case, knowledge representation and inference engine can be designed without explicitly mentioning uncertainty. For the same objective, if we would have designed DSS (Decision support system), then uncertainty measures need to explicitly mention.

### 5.18.4 Version management

As discussed in our NNESDLC, our neural network based expert system for evaluating motivational strategies has been developed using prototyping and spiral model. We created different version with different features before we came out with full-fledged version of the expert system. The table 5.2 summarises the versions and the description of the system.

<table>
<thead>
<tr>
<th>Version</th>
<th>Description of system, features and functionalities</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>Proof of concept for back propagation algorithm with pattern recognition of numbers.</td>
</tr>
<tr>
<td>1.1</td>
<td>Expert system prototype with one motivational strategy and 10 records in the training data set implemented in C++.</td>
</tr>
<tr>
<td>2.0</td>
<td>Expert system prototype with two motivational strategies and 200 records in the training data set implemented in MATLAB.</td>
</tr>
<tr>
<td>3.0</td>
<td>Expert system prototype with two motivational strategies and 560 records in the training data set with parameter settings in algorithm; implemented in MATLAB.</td>
</tr>
<tr>
<td>4.0</td>
<td>Expert system prototype with different activation functions, momentum rate and learning rate</td>
</tr>
<tr>
<td>5.0</td>
<td>Expert system prototype for evaluating all 47 motivational strategies with</td>
</tr>
</tbody>
</table>
600 records in the training data set

6.0 Expert system prototype with more than 2000 records in the training data set

6.1 Expert system prototype with explanation facility

6.2 Expert system prototype with knowledge update facility

6.3 Expert system prototype with batch processing for learning of 47 motivational strategies.

6.4 Expert system prototype where user will select input variables.

If in the future, if our neural network based expert system evolves, then we can create the next version to add more features and more functions.

**Chapter Conclusion**

The chapter has helped us to select various parameters of back propagation algorithm for our neural network based expert system and provide understanding of model of neural network. During the process, we discussed our knowledge engineering and system development life cycle for our neural network based expert system. We also explored the various issues like risk management and version management related to project management.

**5.19 References**


Design of expert system prototype for analysing and structuring motivational strategies on ICT human resources to reduce employee turnover ratio


