CHAPTER 4

PROPOSED EMOTION RECOGNITION SYSTEM

The focus of this chapter is the mathematical framework and the model of the system, which is put forth in the form of architectural diagrams, flowcharts and algorithms relating the processing logic of system.

4.1 THEORETICAL ANALYSIS AND MODELLING

A fuzzy set is defined mathematically by assigning to each member in the set a value representing its grade of membership in the set. These membership values indicate their compatibility with the concept represented by the set and are represented by real numbers in the range [0,1]. The membership function of a fuzzy set A is denoted by μ_A where

$$\mu_A: X \rightarrow [0, 1]$$

Fuzzy sets can be effectively used to represent vague concepts expressed in natural language through fuzzy sets. Fuzzy sets representing linguistic concepts (low, medium, high) are used to define the states of a variable which is termed as fuzzy variable. They are defined by membership functions denoted as $[T_1, T_2] \rightarrow [0, 1]$. Graphs of these functions have *trapezoidal shapes.* Construction of membership functions, operations on them, fuzzy inference rules and other entities in fuzzy set theory can be constructed using neural networks. This is based on learning patterns from sample data.

The two standard fuzzy set operations used in the research are *intersection* and *union.* For two given fuzzy sets A and B

$$(A \cap B)\ (x) = \min[A(x), B(x)],$$

$$(A \text{ U } B)\ (x) = \max[A(x), B(x)]$$

where min denotes minimum and max denotes maximum operator.

Fuzzy variables are variables whose states are fuzzy numbers. When these fuzzy numbers represent linguistic concepts like very small, small, medium etc., in a specific context then they are termed as linguistic variables. The states of each linguistic variable are expressed by linguistic terms interpreted as specific fuzzy numbers. Fuzzy variables provide gradual transitions between states and can deal with observation and measurement uncertainties. Hence they are more suitable for reality when compared with crisp variables.

Inference rules are the various forms of tautologies that are used to make deductive inferences. The truth or falsity of a fuzzy proposition is a matter of degree expressed by a number in the unit interval [0, 1]. The propositions that are employed in the proposed system are "conditional and unqualified" propositions which are expressed by the canonical form

P: If x is A, then y is B.

where x and y are variables whose values are in sets X, Y respectively and A,B are fuzzy sets on X, Y respectively. "x is A" is called the antecedent or premise and "y is B" is called the consequence or conclusion.

<x, y> is R, where R is a fuzzy set on X x Y which is determined for each x Є X and each y Є Y given by the formula

R(x, y) = f[A(x), B(y)], where f denotes a binary operation on [0, 1] representing a fuzzy implication.

Rule-based inference system can be mapped into a neural network. Firstly, the data variables are assigned input nodes, target concepts are assigned output nodes and intermediate concepts are assigned hidden units. The initial domain rules determine how the inputs and outputs are linked and how the links are weighed.

The fuzzy inference system is a computing framework based on the concepts of fuzzy set theory, fuzzy if-then rules and fuzzy reasoning. It has three conceptual components: a rule base containing the fuzzy rules, database which defines the membership functions used in the fuzzy rules and a reasoning mechanism which performs the inference procedure. The three main fuzzy inference systems are: Mamdani fuzzy models, Sugeno fuzzy models and Tsukamoto fuzzy models. The Mamdani fuzzy model has two variations: single-input single-output model and two-input single-output model. The latter is used in the Neuro-Fuzzy system of the proposed system.

One of the successful application areas of fuzzy systems is: fuzzy control wherein fuzzy controllers or fuzzy inference systems are used

The Neuro-Fuzzy system (Fuzzy Controller, as depicted in Fig 4.1) is the core of the emotion recognition system and has four modules :

1. Fuzzification Module: converts crisp inputs into appropriate fuzzy sets represented in the fuzzy database.
2. Fuzzy Inference Engine: evaluates the fuzzy rules stored in the fuzzy rule base.
3. Fuzzy Rule Base: consists of fuzzy inference rules which formulate the knowledge of the problem.
4. Defuzzification Module: converts fuzzy set into a single crisp value.

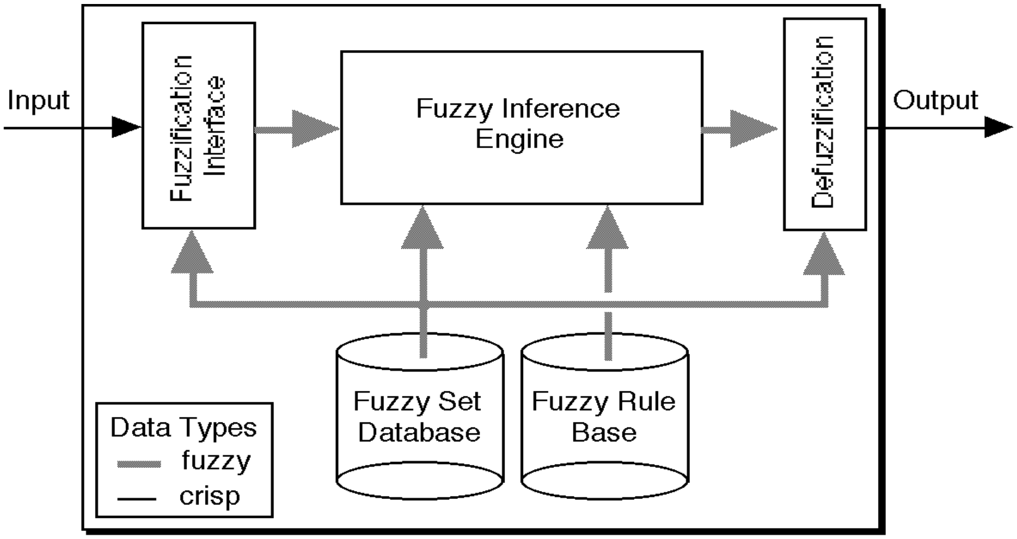


Fig 4.1: Fuzzy Controller

The functionality can be expressed as a cycle of four steps :

1) Measurement of input variables and specification of output variables.
2) Convert the input, crisp values into fuzzy sets i.e., fuzzification.

3) Inference engine uses fuzzy sets to evaluate control rules stored in the fuzzy rule base and generates a fuzzy set.
4) The fuzzy set is converted into a single crisp value which represents it i.e., defuzzification.

The actions taken by the fuzzy controller in individual control cycles are represented by the defuzzified values. Fuzzy inference rules are the set of control rules which use fuzzy propositions (i.e., linguistic descritptions).

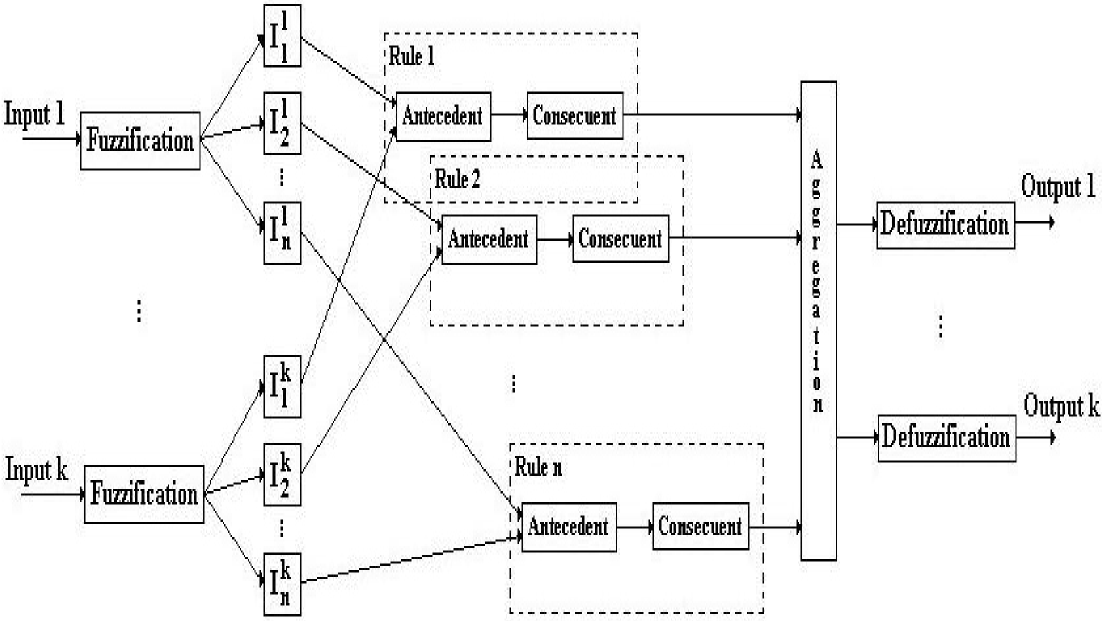


Fig 4.2: Functional Diagram of Fuzzy Controller

Step 1: Identify the relevant I/O variables and ranges of their values and select meaningful linguistic states for each variable and express them by appropriate fuzzy sets.

The system considers the variables taken over the set of six emotions i.e.,

{Happiness, Despair, Anger, Fear, Disgust, Surprise}

Input variables :

The first input variable is the emotional weight of the keyword denoted as Weight, W where the range of W is [0, 20] as represented in Fig 4.2.

Linguistic states of W = { low, medium, high }

Range of low is 0-10, Range of medium is 5-15, Range of high is 10-20

The ranges are on X-Axis and membership values of 0-1 on the Y-Axis.

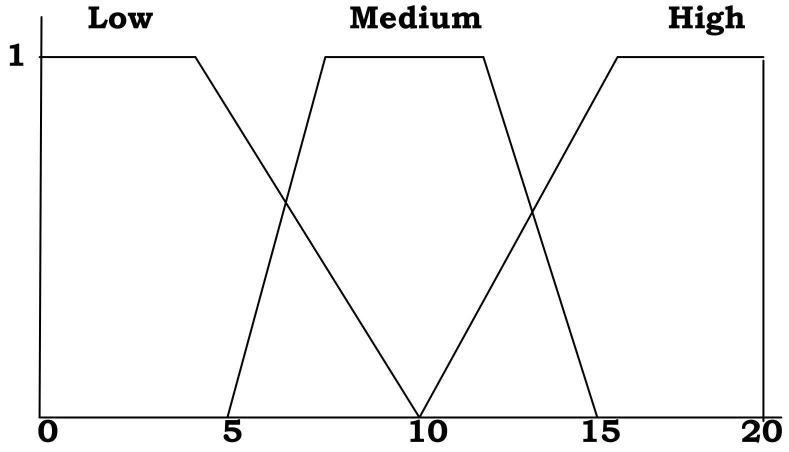


Fig 4.3: Membership function for the variable WEIGHT, W

The second input variable is the polarity of the emotional keyword denoted as Polarity, P where the range of P is [-1, 1] as in Fig 4.3.

Polarity has two linguistic states { positive, negative }

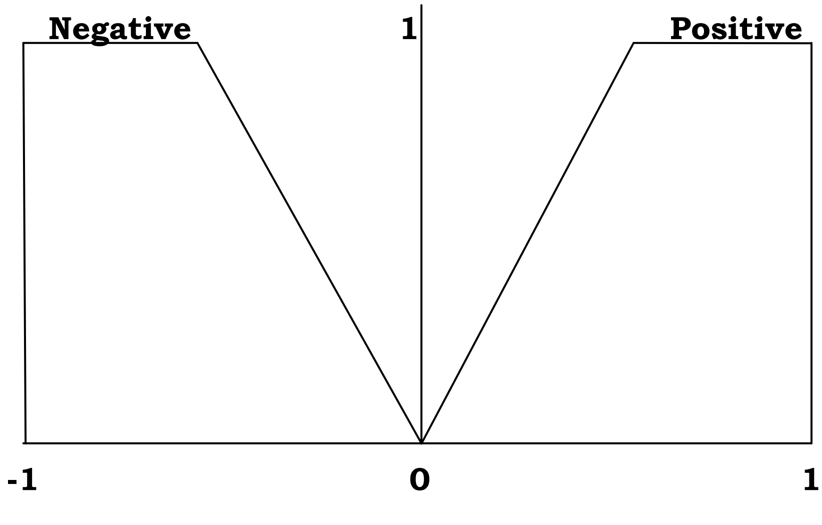


Fig 4.4: Membership function for the variable POLARITY, P

Output variable :

The system represents the emotion of the event through the output variable, E where the range of E is [-20, 20] as in Fig 4.3.

The variable emotion has six linguistic states { positive low, positive medium, positive high, negative low, negative medium, negative high } with ranges as shown in Fig 4.4.

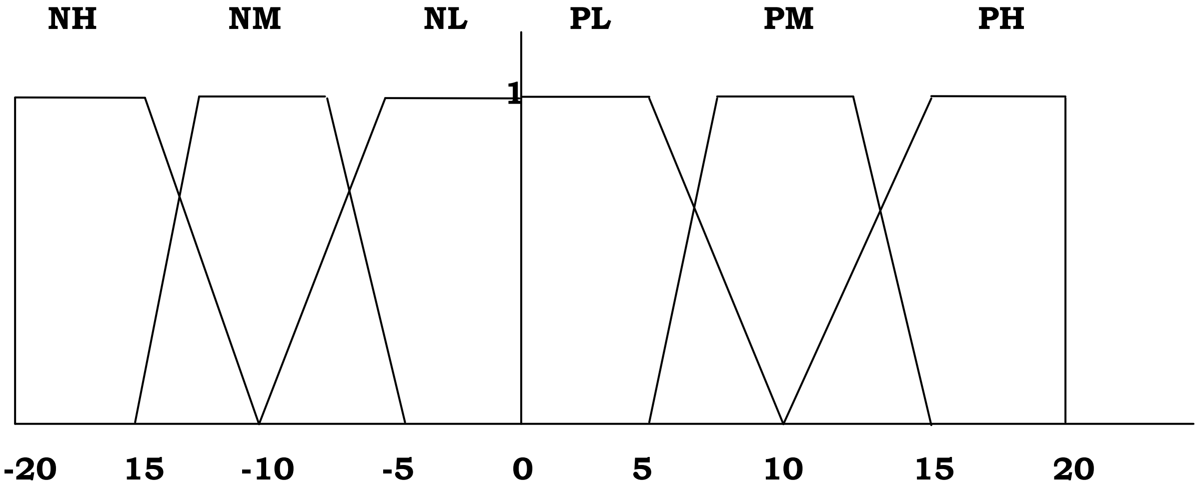


Fig 4.4: Membership function for the variable EMOTION

Step 2: A fuzzification function is defined for each input variable to express the associated measurement uncertainty. The function interprets measurements of input variables, each expressed by a real number.

Fuzzification function :

1) f_w: [x, y] → R i.e., f_w : [0, 20] → R

2) f_p: [a, b] → R i.e., f_p : [-1, 1] → R

Step 3: The knowledge of the system is formulated as a set of fuzzy inference rules, which are obtained from empirical data using suitable learning methods, with the help of a neural network. The rules are are framed over the linguistic states of the input and output variables.

The fuzzy inference rules are formulated as follows :

1. *If weight is low and polarity is positive then emotion is positive low.*
2. *If weight is medium and polarity is positive then emotion is positive medium.*
3. *If weight is high and polarity is positive then emotion is positive high.*
4. *If weight is low and polarity is negative then emotion is negative low.*
5. *If weight is medium and polarity is negative then emotion is negative medium.*
6. *If weight is high and polarity is negative then emotion is negative high.*

		Polarity	
	Emotion	Positive	Negative
Weight	Low	Positive Low	Negative Low
	Medium	Positive Medium	Negative Medium
	High	Positive High	Negative High

Table 4.1: Inference Matrix

Step 4: Input variables of a fuzzy controller (see Fig 4.5) shpold be measured and properly combined with relevant fuzzy information rules to make inferences regarding the output variables. This is the task of the inference engine. Here , firstly the given fuzzy inference rules are converted into equivalent simple fuzzy conditional propositions of the form : If < w, p > is A x B then e is C where [A x B] (x, y) = min[A(x), B(y)]

for all $x \in [\ 0, 20\ ]$ and $y \in [-1, 1]$

When the fuzzy rule base consists of n fuzzy inference rules the reasoning schema is as follows :

Rule 1 : If < w, p > is $A_1 X B_1$ *then e is* C_1*.*

Rule 2 : If < w, p > is $A_1 X B_2$ *then e is* C_2*.*

Rule 3 : If < w, p > is $A_2 X B_1$ *then e is* C_3*.*

Rule 4 : If < w, p > is $A_2 X B_2$ *then e is* C_4*.*

Rule 5 : If < w, p > is $A_3 X B_1$ *then e is* C_5*.*

Rule 6 : If < w, p > is $A_3 X B_2$ *then e is* C_6*.*

Fact : < w, p >. is $f_w(x_0) X f_p(y_0)$

Conclusion : e is C.

The symbols A_j, B_j, C_j (j = 1, 2, ……… n) denote fuzzy sets that represent the linguistic states of variables w, p, e. The linguistic states of <w, p> are fuzzy sets defined on [0, 20] X [-1, 1] which are calculated from their counterparts for w and p by the minimum operator. Inference engine is a neural network with two inputs , one hidden layer and one output. The activation function is the trapezoidal function in each neuron.

$$\mathrm{trapezoid}(x;a,b,c,d)=\max\left(\min\left(\frac{x-a}{b-a},1,\frac{d-x}{d-c}\right),0\right)$$

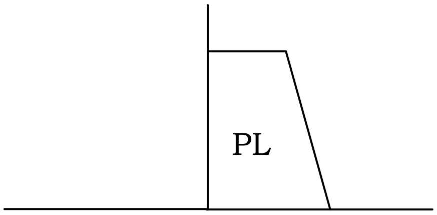
The parameters {a, b, c, d} (a<b<=c<d) determine the x coordinates of the four corners of the trapezoidal membership function.

Step 5 : A suitable defuzzification method must be selected which converts each conclusion obtained by the inference engine, expressed in terms of a fuzzy set, to a single real number. The set to be defuzzified for any input measurement w = x_0 and p = y_0 is the set C. The various defuzzification methods are Center of Area Method, Center of Maxima Method and Mean of Maxima method. The Mean of Maxima method is used here for defuzzification wherein the defuzzified value, d_{MM} (C) is the average of all values in the crisp set M = { z_k / C(z_k) = h (C) }

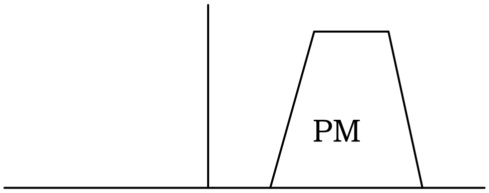
$$d_{MM}\ (\ C\ )\ =\ \sum_{z_k \in M}\ (\ z_k\ /\ |\ M\ |\ )$$

Fuzzy Rule Base : The fuzzy rules stated are schematically represented as follows:

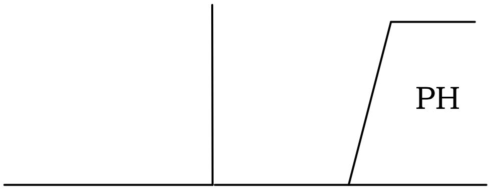
R1 :



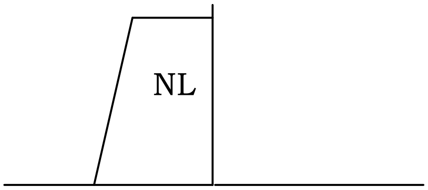
R2 :



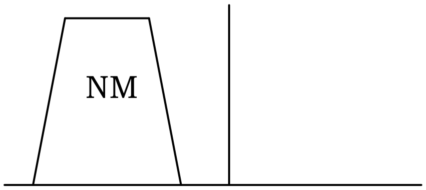
R3 :



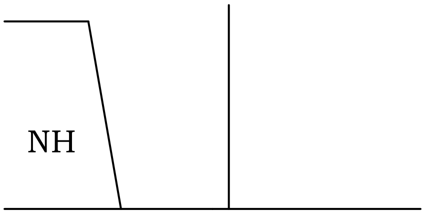
R4 :



R5:



R6 :



A fuzzy system can be approximated with an appropriate neural network as it has the inherent capability to perform massive parallel processing of information and learning capabilities of neural network which can be utilized for modifying fuzzy inference rules of the system on the basis of experience.

The trapezoid function is chosen as the activation function for each neuron and the backpropagation algorithm is used for training the neural network which maps a set of given input patterns into a set of required output patterns.

Suppose that the neural network has ‘q’ neurons in the hidden layer. According to the available heuristic rules, $q \geq 6$. Inputs of the neural network are partitioned into two subsets that correspond to variables ‘w’ and ‘p’. A fuzzy number is assigned to each variable which represents a particular linguistic state of the respective variable. Each output is allocated to one of the equally distributed discrete points in the range [-20, 20] ⇒ -20, -19, -18,……….. 0……….18, 19, 20. Its value specifies, at the given discrete point in the range [-20, 20], the membership grade of the fuzzy number that represents the output variable ‘e’. Each fuzzy inference rule is represented in the neural network by an input vector that specifies the linguistic states of the input variables ‘w’ and ‘p’ and output vector that defines the related fuzzy number of the output variable ‘e’.

Ex : The fuzzy inference rule : If ‘w’ is low and ‘p’ is negative then ‘e’ is negative low; is represented in the neural vector by the following vectors:

Input Vector: < 1, 0, 0 ; 1, 0 >

Output Vector: < 1, 0, 0, 0, 0, 0 >

Values can be between 0 and 1 depending on membership values.

These two vectors represent one input-output pair of a training set . The full training set consists of all 6 fuzzy inference rules. The neural network gradually learns to associate the proper fuzzy number at the output with each of the possible pairs of input linguistic states, when this training set is applied to the backpropagation algorithm,

After the training process, the neural network responds correctly to every input vector in the training set. To employ the network for generating appropriate control actions, it must be augmented with fuzzification and defuzzification modules. The fuzzification module feeds appropriate input values into the neural network in the form of the input vector. Then the neural network produces an output vector which defines the fuzzy set representing the conclusion of the inference. Error function E_p, employed in the backpropagation learning algorithm in a fuzzy neural network with 'm' outputs for each training sample 'p' is defined by the formula

$$E_p = ½ \sum (T_k^p - Y_k^p)^2$$

T_k^p is the target output and Y_k^p is the actual output for the training sample p. E_p is calculated using fuzzy arithmetic. The stopping criterion for fuzzy neural network must also be fuzzified. Suppose that $T_k^p = Y_k^p$ for all k, represents a perfect match of the actual outputs with the target outputs. Then, the algorithm can be terminated.

4.2 PROPOSED SYSTEM ARCHITECTURE

The system comprises of the following units as shown in Fig 4.6

1. Input Unit.
2. Event Processor.
3. Emotional State Calculator (Neuro Fuzzy System as in Fig 4.7) with the following five layers :
    a) Input Layer
    b) Input Fuzzy Layer
    c) Conjunction Layer
    d) Output Fuzzy Layer
    e) Output Layer
4. Output Unit.

The architecture is designed such that every unit is well-defined and self-contained. The processing of the Input Unit, Event Processor and Output Unit are implemented through JAVA programs. The Emotional State Calculator is the Neuro-Fuzzy System (Mamdani model) developed in MATLAB. The Neuro-Fuzzy System takes its inputs from the Event Processor and sends the outputs to the Output Unit.

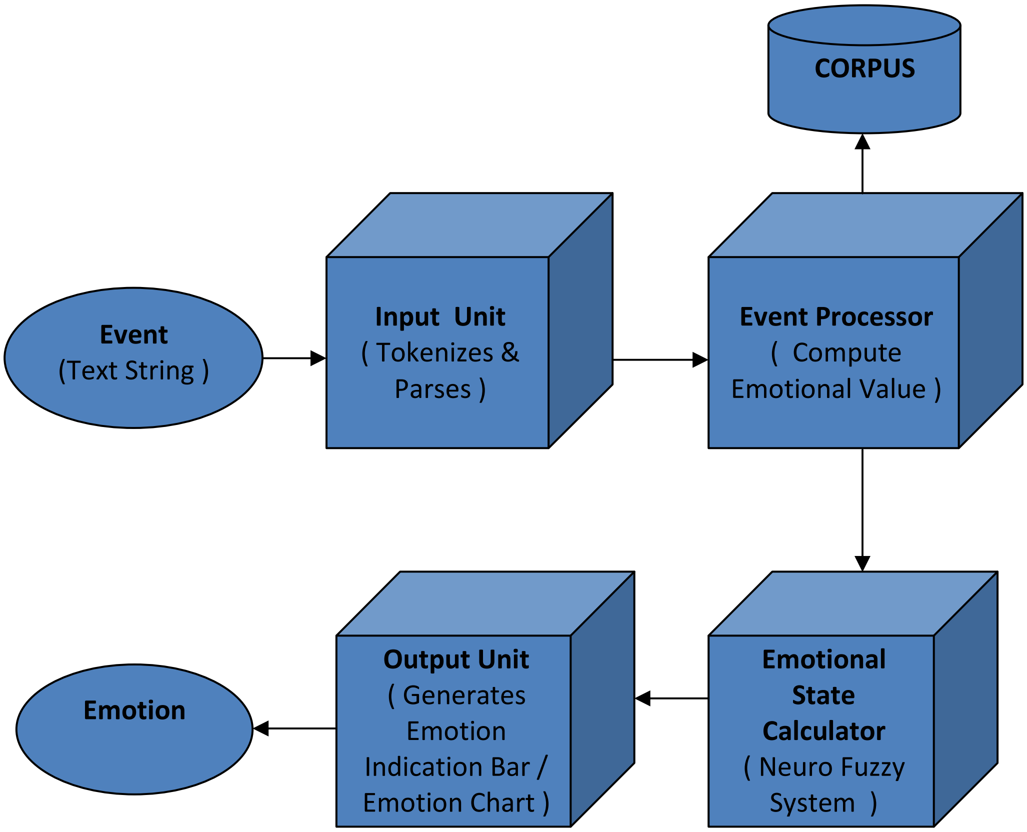


Fig 4.6: System Architecture

***Input Unit*:** This unit accepts the string representing the real world event, parses it and divides into tokens .

Event Processor: The tokens are compared to a corpus of emotional keywords (WordNetAffect), and the emotional values are computed which are then converted into linguistic variables.

***Emotional State Calculator*:** Depending on the category of emotion recognized by the event processor, the corresponding network is selected as depicted in Fig 4.7. There are six networks for six emotions whose basic functionality is the same.

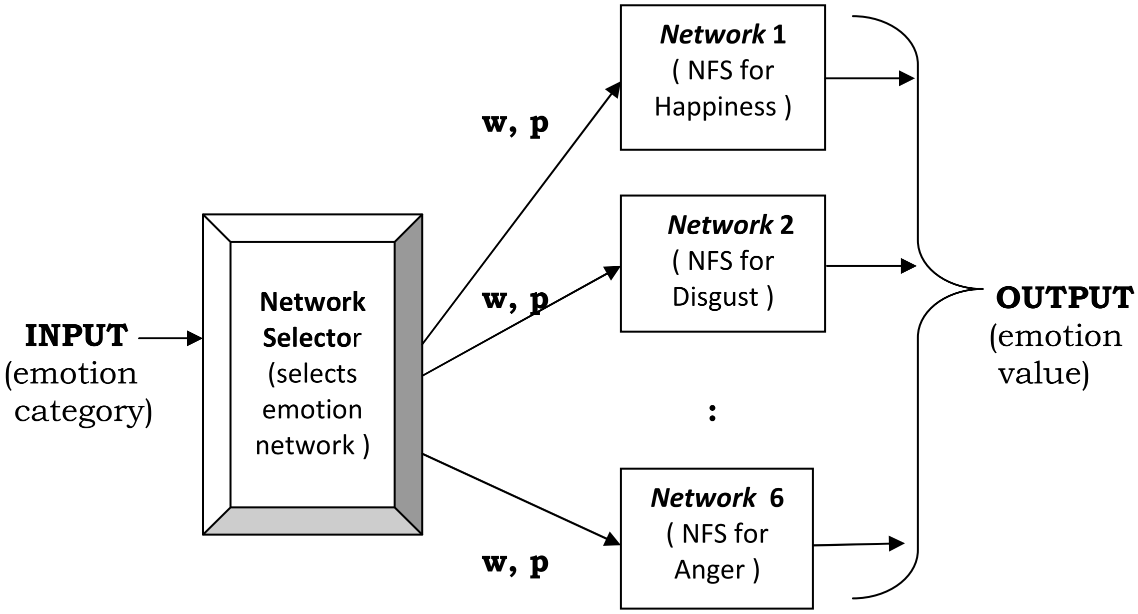


Fig 4.7: Emotional State Calculator

Each network is a neuro-fuzzy system which takes the variables, emotional weight (w) and polarity(p) as input, fuzzifies them, processes and generates the fuzzified value for emotion which is later defuzzified and value for emotion is generated as output.

Output Unit : The value generated by the above unit is mapped onto the corresponding level of intensity by the program and the output is generated in the form of an emotion indication bar displaying the emotion and its intensity

4.3 RESEARCH METHODOLOGY

The procedural details of the system are elaborated as follows :

Phase 1: The input to the system is a string representing the real world event. The string is accepted, parsed and divided into tokens by the Input Unit.

Phase 2: The Event Processor considers the tokens of the string and compares them both to a corpus of emotional keywords (WordNetAffect) and a lexicon of the English language . The comparison is done using the Knuth-Morris-Pratt pattern matching algorithm [30]. The emotional values and the polarity of the emotional keywords of the event are computed as explained in the algorithm of *Recognition Task*. The emotional weight and polarity values are converted into corresponding linguistic variables of W and P and fed to the emotional state calculator which is a neuro-fuzzy system.

Phase 3: The network selected by the selector for a specific emotion is a neuro-fuzzy system, the structure of which is given in Fig 4.8. It takes the crisp variables, emotional weight (w) and polarity(p) as input, fuzzifies them, processes and generates the fuzzified value for emotion which is later defuzzified and the value for emotion is generated as output. As discussed earlier, the emotional weight is fuzzified based on the linguistic states of low, medium and high; and the polarity is fuzzified based on the linguistic states of positive and negative; using the fuzzificaion function. The inferences are made using the rules (R1…R6) and the linguistic state for the resultant emotion is generated. This is defuzzified and the value for the emotion is given as output.

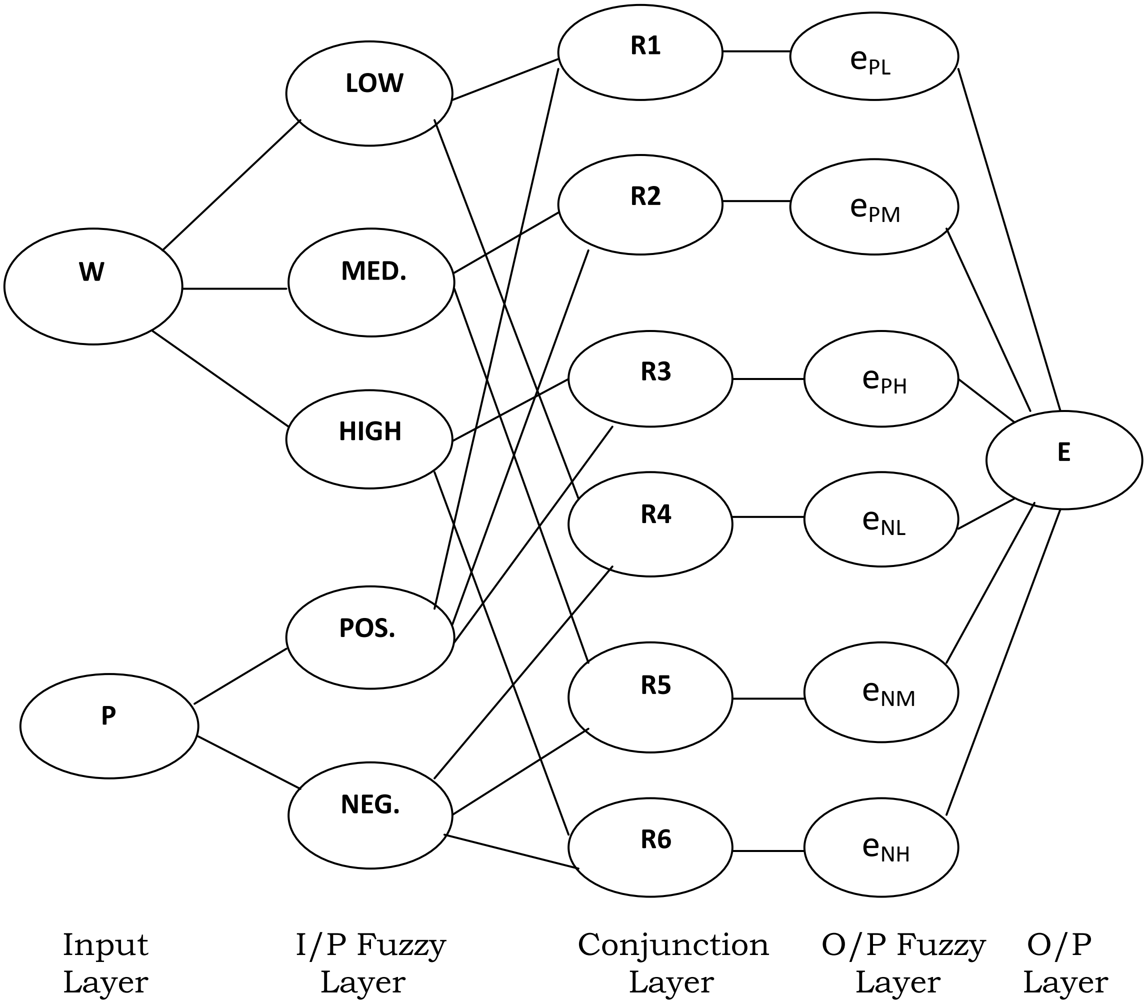


Fig 4.8: Neuro-Fuzzy System for each emotion

In the Neuro-Fuzzy System the activation level of an input unit is the value of a certain input variable in the given instance. The input value is passed onto fuzzy set units, which then translate the value into a degree of membership as the activation level of a fuzzy set unit. The conjunction unit will take the minimum of the inputs (degrees of membership) it receives from the input fuzzy set units below. Now, the output fuzzy set unit will collect information from one or more conjunction units (each corresponding to a fuzzy rule). An output fuzzy unit takes the maximum of its inputs. Then the output unit generates the final result by integrating the information from the output fuzzy set units.

The activation function used in each layer is as follows :

1. The input layer : no activation function.
2. The input fuzzy layer : the trapezoidal function.
3. The conjunction layer : the min function.
4. The output fuzzy layer: the mean-of-maxima function.
5. The output layer: no activation function.

Phase 4: The value generated by the above unit is mapped onto the corresponding level of intensity by the program and the output is generated in the form of an emotion indication bar displaying the emotion and its intensity. In case of multiple emotions an emotion chart is displayed.

The procedure stated in this section is given in the form of algorithms corresponding to the processing logic of the programs. The corresponding flowcharts increase the level of detail.

ALGORITHM 1: *RECOGNITION TASK*

1. *Input the event string.*
2. *Divide the string into tokens using the tokenizer.*
3. *Compare the tokens to the database of emotional keywords using Knuth-Morris-Pratt pattern matching algorithm.*

4. *If the tokens do not match any keyword then generate output as NEUTRAL.*

5. *For the tokens that match do the following operations:*

   a. *Store the emotional keywords and category in a table.*

   b. *NOC = no.of categories of emotional keywords.*

   c. *Compute the emotional value of the keyword as follows :*

      i. *Use the K-Window1 algorithm to search the 'k' words preceding the seed.*

         Case 1 : If the seed is not preceded by any emotion modifier then assign a grade of 1.

         Case 2 : If the seed is preceded by one modifier then assign a grade of 2.

         Case 3 : If the seed is preceded by more than one modifier then assign a grade of 3.

      ii. *Count the frequency of the keyword.*

      iii. *Compute the emotional-value of the keyword as follows :*

$$emoval = tf_{i,j} * itf\ (k_i) * scaling\text{-}factor$$

where $tf_{i,j}$ *= term frequency, is the raw frequency of emotional keyword* k_i *of a emotional category* e_j*.*

itf = inverse term frequency is the logarithm of n_i *(NOC) over N (total no.of emotional categories) scaling-factor = 5, in accordance with the ranges considered.*

```
        iv. Calculate  weight  =  grade *  emotional-value.
            If (weight > 0 and weight <=5) then intensity = low.
            If (weight >7 and weight <=12) then intensity = medium.
            If (weight >15 and weight <=20) then intensity = high.
            If (weight > 20) then intensity = high. (Normalize value)
    d.  Compute the polarity of the keyword as follows :
        i.  Use the K-Window2 algorithm to detect occurrence of negation words before the seed.
            Case 1: If the seed is preceded by a negation word  then polarity is negative  i.e., -1
            Case 2 : If the seed is not preceded by a negation  word then  polarity is positive i.e., 1.
            Similarly determine the polarity by detecting negation words after the seed.
6.  Input the category to the NeuralNetwork Selector.
7.  Input the weight and polarity to the specific network selected.
8.  Evaluate using the inference rules of Neuro-Fuzzy System.
9.  Generate the emotion in the form of “Emotion Indication Bar”.
10. Generate Emotion Chart for events having more than one emotion by repeating step(5) for each category.
```

The Knuth-Morris-Pratt algorithm for a string T (emotional keyword from corpus with n characters) and a string P (token from input string) is :

Algorithm 2: *KMPMatch (T, P)*

```
{ f = KMPFailureFunction(P)
    i=0, j=0
    While ( i<n ) do
   {   If P[j] = T[i] then
            If ( j = m-1 ) then return i-m+1
           i = i + 1, j = j+1
       else if ( j > 0 ) then j = f(j)-1
       else i = i+1
    }
   Return " No match found "
}
```

Algorithm *KMPFailureFunction(P)*

```
{  i=1, j=0, f(0) = 0
   While ( i<m ) do
           If P[j]=P[i] then
              f(i) = j+1
              I = i+1, j = j+1
         else if ( j>0) then j = f(j)-1
         else { f(i)=0, i = i+1  }
}
```

The flowchart for recognition task as shown in Fig 4.9 depicts the parsing, tokenizing of the input string; comparison with emotional keywords and calculation of emotional values followed by identification of emotion and its intensity.

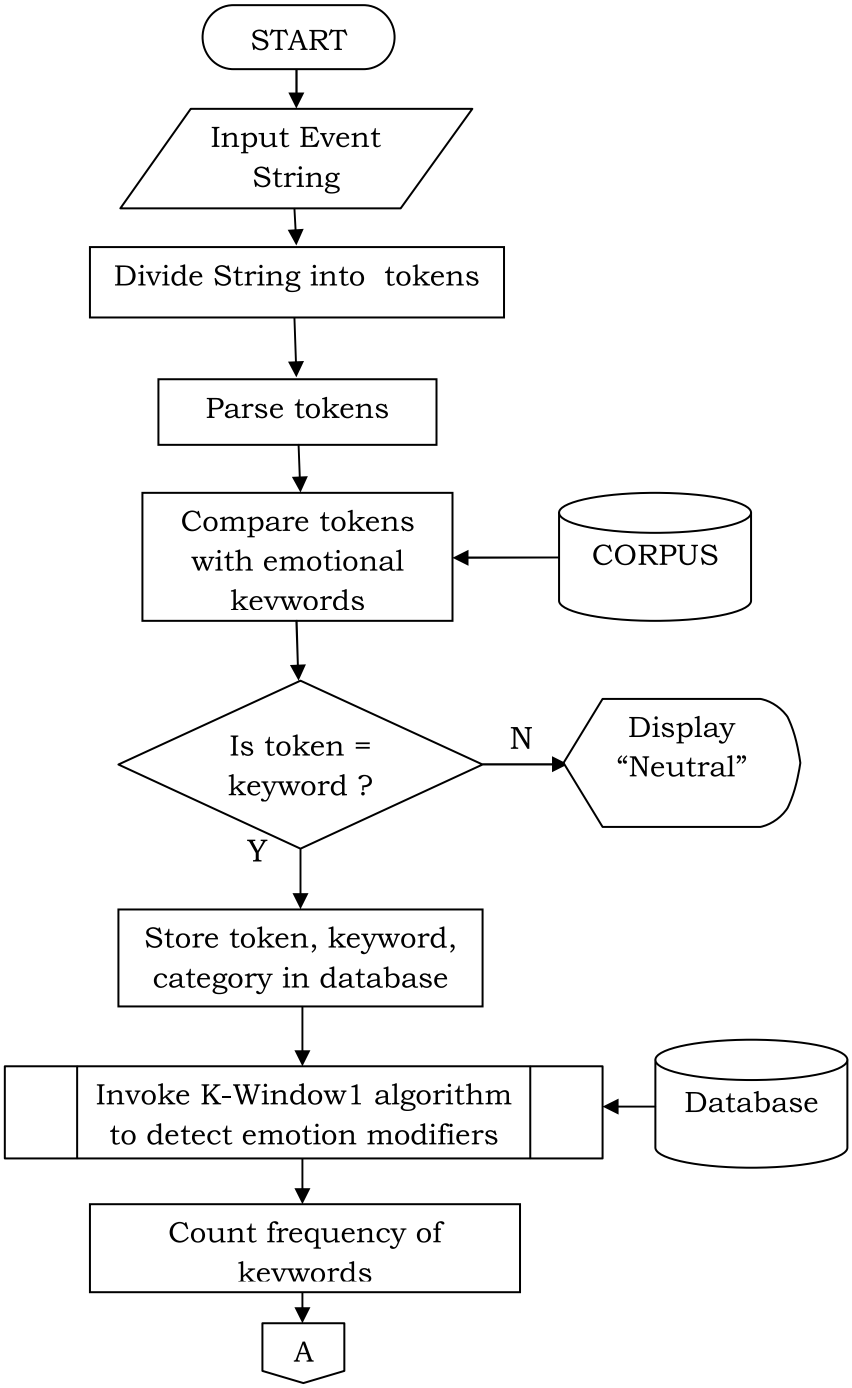

START
Input Event String
Divide String into tokens
Parse tokens
Compare tokens with emotional kevwords
CORPUS
Is token = keyword ?
N
Display "Neutral"
Y
Store token, keyword, category in database
Invoke K-Window1 algorithm to detect emotion modifiers
Database
Count frequency of kevwords
A

Fig 4.9: Flow Chart for Recognition Task of the System (Cont….)

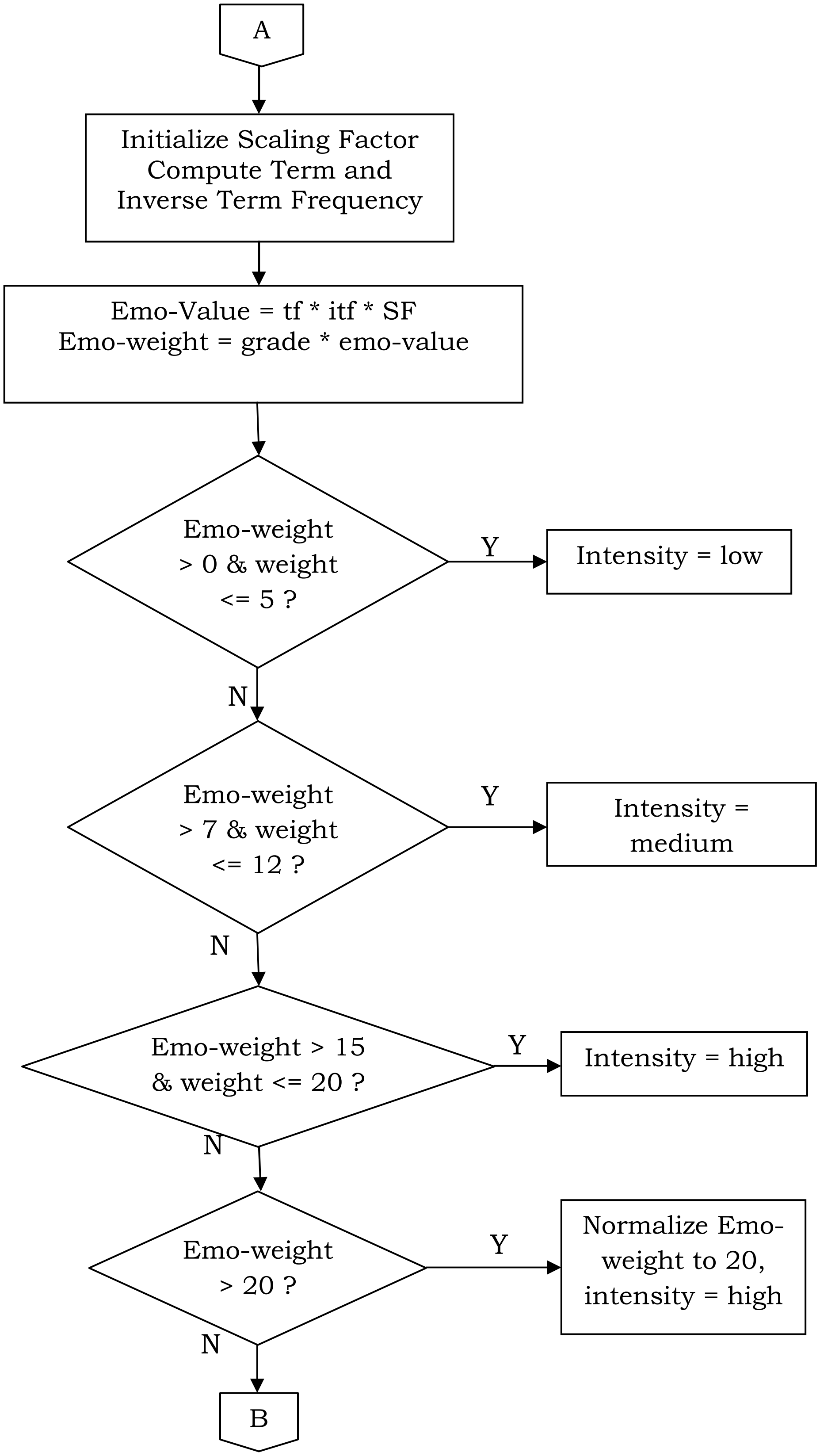


Fig 4.9: Flow Chart for Recognition Task of the System (Cont.....)

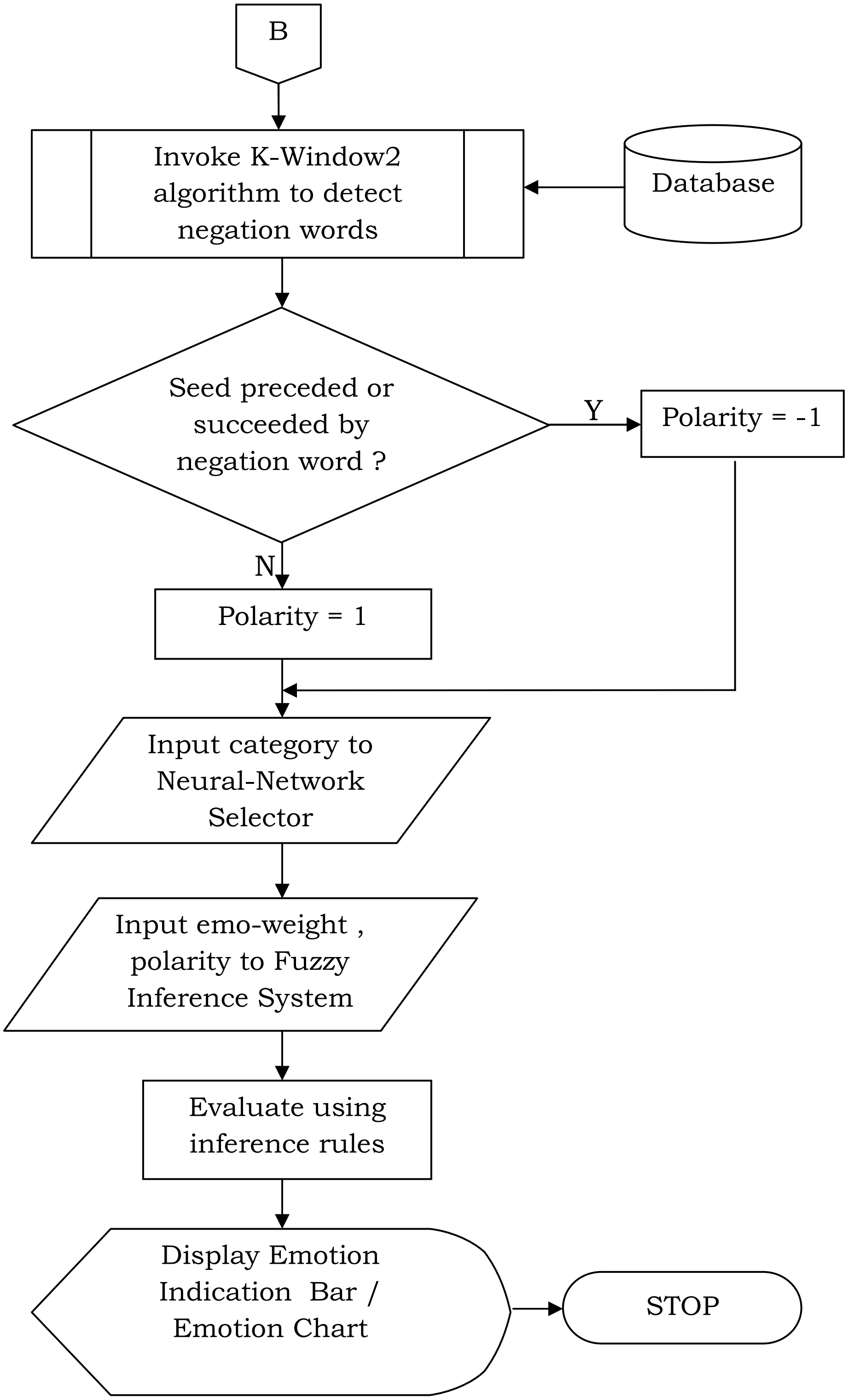

B
Invoke K-Window2 algorithm to detect negation words
Database
Seed preceded or succeeded by negation word ?
Y
Polarity = -1
N
Polarity = 1
Input category to Neural-Network Selector
Input emo-weight , polarity to Fuzzy Inference System
Evaluate using inference rules
Display Emotion Indication Bar / Emotion Chart
STOP

Fig 4.9: Flow Chart for Recognition Task of the System

Algorithm 3: *K-Window1(S) - To compute the grade of keyword using emotional modifiers, emod[ ].*

```
1. Count no.of words(NoW) in input string, S.
2. Find the position (wordno) of the emotional keyword in S.
3. For i = 1 to wordno – 1
   {  count = 0
      For j = 1 to n
          If word[i] = emod[j] then count = count+1
      If ( count =0 ) then grade = 1
      else if (count =1) then grade = 2
      else if (count >=2 ) then grade = 3
   }
4. Return grade.
```

The list of emotional modifiers (ex: very, extremely etc.) are manually defined and stored in the database. The number of modifiers preceding the seed element indicate the intensity of the emotion in the event string.

The flowchart for K-Window1 algorithm portrays the procedure to detect the occurrence of emotion modifiers preceding the emotional words which determine the intensity of the emotion.

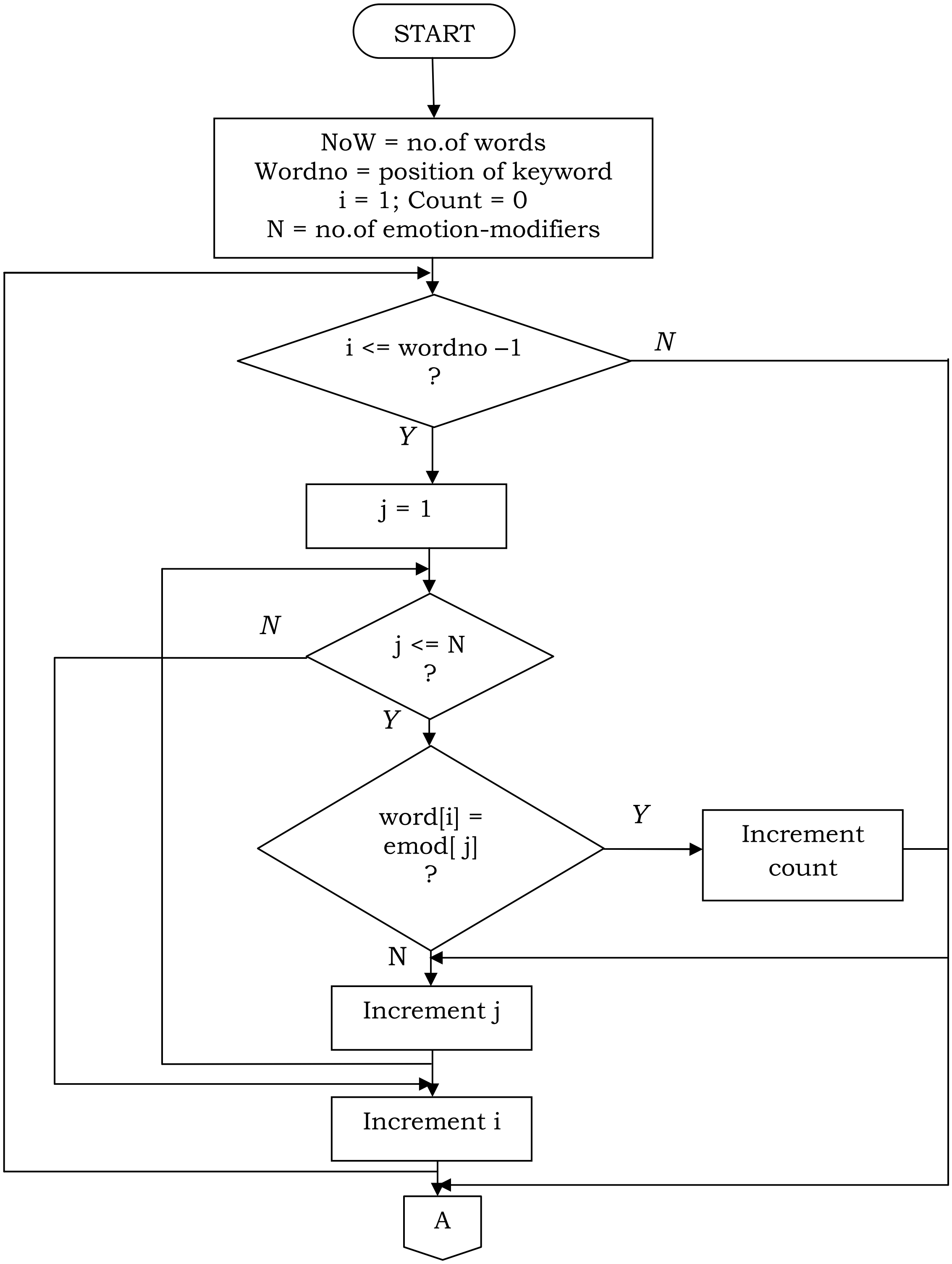


Fig 4.10: F/C for K-Window1 algo. to detect emotional modifiers (cont...)

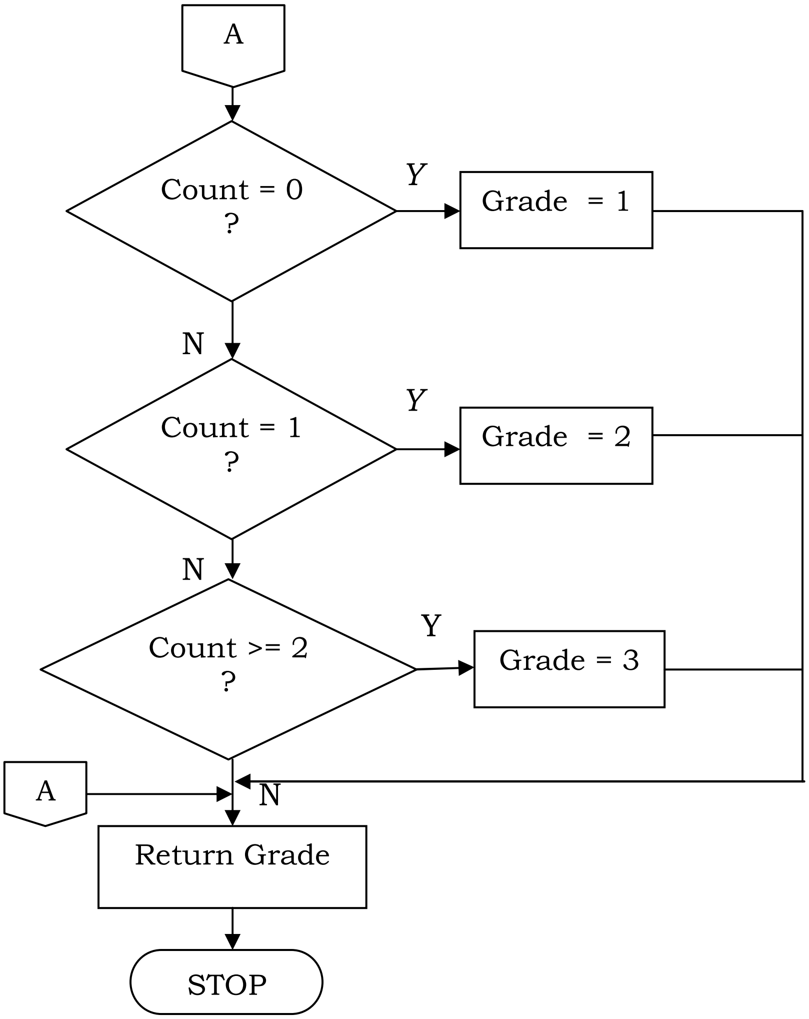


Fig 4.10: Flowchart for K-Window1 algo. to detect emotional modifiers

***Algorithm 4**: K-Window2(S) - To compute the polarity of keyword using negation words, neg[ ].*

```
1. Count no.of words(now) in input string, S.
2. Find the position (wordno) of the emotional keyword in S.
3. For i = 1 to wordno – 1
   {  c = 0
      For j = 1 to n
          If word[i] = neg[j] then c = c+1
      If ( count >= 1 ) then { polarity = -1, GOTO step(5) }
                    else { polarity = 1, GOTO step(5) }
4. For  i =  wordno + 1 to now
   {  c = 0
      For j = 1 to n
          If word[i] = neg[j] then c = c+1
      If ( c >= 1 ) then polarity = -1
                    else polarity = 1
5. Return polarity.
```

The flowchart for the training task compares the emotion and intensity of the event generated by the NFS with that of the inter-annotator agreement and then invokes the training algorithm accordingly.

The flowchart for K-Window2 algorithm depicts the detection of negation words which decide the polarity (+ve / -ve) of the emotional keyword.

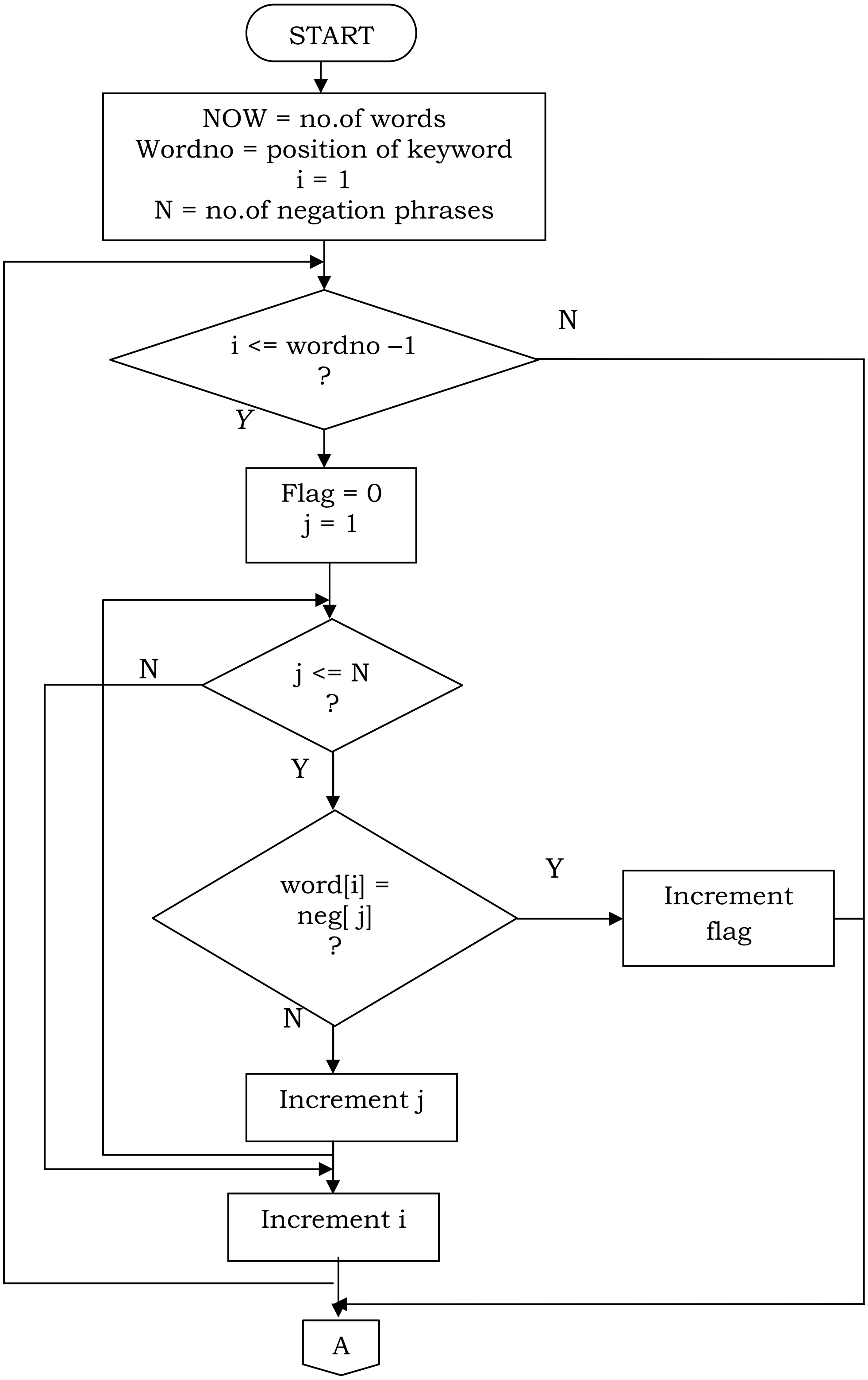


Fig 4.11: F/C for K-Window2 algorithm to detect negation words (cont...)

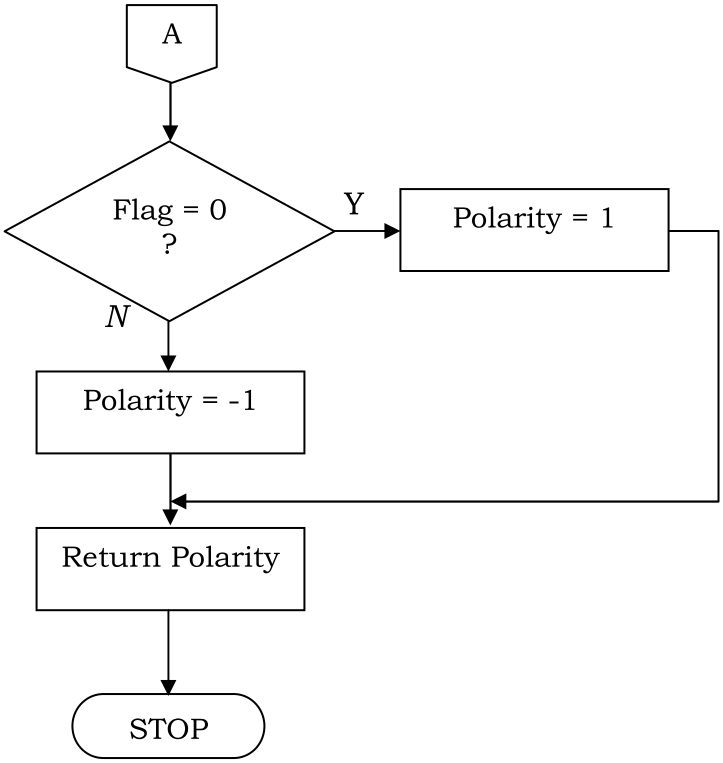


Fig 4.11: Flowchart for K-Window2 algorithm to detect negation words

ALGORITHM 5 *: Training Task*

1. *Display the event string to the Annotator.*
2. *Display the set of predefined emotions and levels.*
3. *Record the values selected by annotator.*
4. *Compare the values with that generated by the system.*
5. *If they match then display output, otherwise include in the training sample. Use the back propagation algorithm to train the Network.*
6. *Make a comparative study to evaluate the performance of the system.*

BACKPROPAGATION ALGORITHM *:*

1. *Initialize weights of the network to a small random number.*
2. *Apply pairs (x, t) of the training set in some order.*
3. *Calculate the actual output y for each x.*
4. *Find the square error as* E_p, $E_p = \frac{1}{2} \sum (Tkp - Ykp)^2$.
5. *Using* E_p *, the weights in the network are updated .*
6. *Calculate the cumulative cycle error as E.*
7. *Compare the cumulative error with the largest acceptable error* E_{max} *at the end of each cycle.*

8. *When E <= E_{max} , a solution is obtained which means that the network represents the desired membership function.*

9. *When E > E_{max} , a new cycle is initiated.*

10. *The algorithm is terminated when a solution is obtained or the no.of cycles exceeds the limit.*

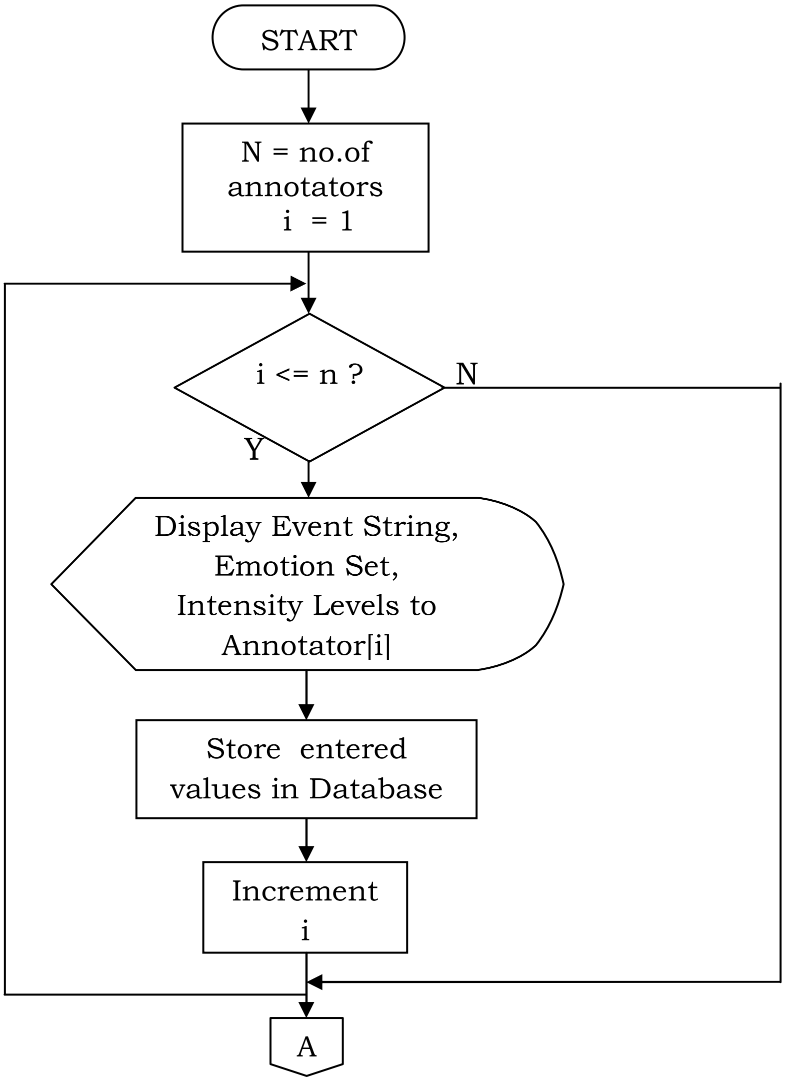

START
N = no.of annotators i = 1
i <= n ?
N
Y
Display Event String, Emotion Set, Intensity Levels to Annotator[i]
Store entered values in Database
Increment i
A

Fig 4.12: Flowchart for Training Task (cont…)

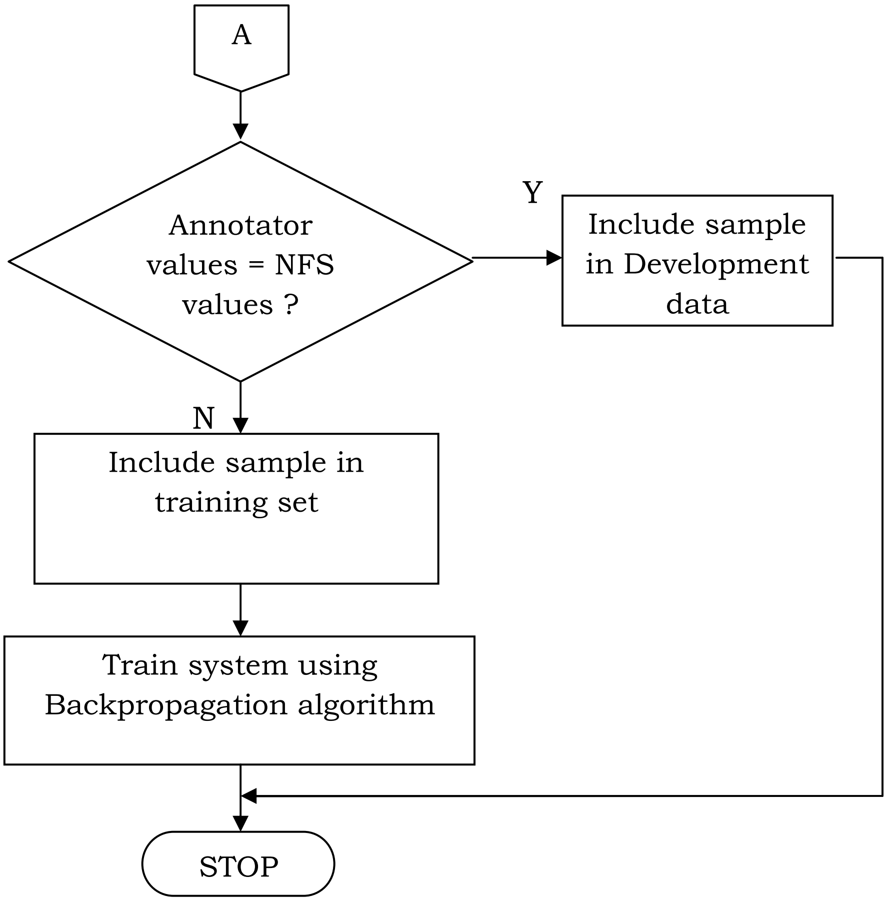

A
Annotator values = NFS values ?
Y
Include sample in Development data
N
Include sample in training set
Train system using Backpropagation algorithm
STOP

Fig 4.12: Flowchart for Training Task