# Chapter 4

**Detailed Design**

**4.1 Structure Chart**

Structure charts are used to specify the high level design or architecture of a computer program. As a design tool, they help the programmer in dividing and conquering a large software problem, i.e. recursively breaking a problem down into parts that are small enough to be understood by a human brain.

The process is called top-down design or functional decomposition. Programmers use a structure chart to build a program in a manner similar to how an architect uses a blueprint to build a house. In the design stage, the chart is drawn and used as a method for the client and various software designers to communicate. During the actual building of the program, the chart is continuously referred to as master plan. Often, it is modified as programmers learn new details about the program. After a program is completed, the structured chart is used to fix bugs and to make changes. Structured chart of the project is shown in figure 4.1 and table 4.1 contains the notations and description.

**4.2 Functional Description of Modules**

* **Main Module**: - The entire module is connected to the main module which performs the sentiment analysis part.
* **Retrieval Module**: - This module’s main function is to retrieve all the tweets from the Twitter corresponding to the given keyword. The input to this module is the keyword and the output is the tweets corresponding to the keyword.
* **Pre-processing Module**: - This module takes tweets as the input from the main module. It removes the rewets.
* **Feature Reduction**: - This module takes tweets as the input from the main module. It replaces URLs with URL keyword, usernames with USERNAME keyword. It removes the slang words. It returns the normalised data to the pre-processing module

i

**Main**

**Feature Reduction**

**Display Module**

a k

b c d g h j

**Compute Probability**

**Retrieval module**

**Training module**

**Pre-processing**

e f l m

**Compute large**

n o p q r s

**Negative Probability**

**Positive Probability**

**Neutral Probability**

Figure 4.1 Structure Chart

**Table 4.1 Symbols and description**

|  |  |
| --- | --- |
| **SYMBOLS** | **DESCRIPTIONS** |
| a | Tweets |
| b | Keyword |
| c | Preprocessed text |
| d | Tweets |
| e | Normalised tweets |
| f | Tweets |
| g | Output training file |
| h | Input training file |
| i | Classified data |
| j | Probability |
| k | Pre-processed twitter data, training data |
| l | Maximum probability |
| m | Positive,negative,neutral probabilities |
| n | Positive probability |
| o | Input file containing tweets |
| p | Negative probability |
| q | Input file containing tweets |
| r | Neutral probability |
| s | Input file containing tweets |

* **Training module**: - This module takes input training file as the input and puts that data into output training file
* **Compute Probability**: - It takes the input as the pre-processed twitter data and the training data and computes the probability for the twitter data. It returns the probability value to the main module.
* **Compute large**: - It takes the input as the positive probability, negative probability and the neutral profanity. It computes the maximum of these probabilities and returns that value.
* **Positive probability**: - It takes the input as the file containing tweets, counts the number of words in that file and compare that thing with positive training data using Naïve Bayes technique and returns the calculated positive probability. Then it returns this value to the compute probability module.
* **Negative probability**: - It takes the input as the file containing tweets, counts the number of words in that file and compare that thing with negative training data using Naïve Bayes technique and returns the calculated negative probability. Then it returns this value to the compute probability module.
* **Neutral probability**: - It takes the input as the file containing tweets, counts the number of words in that file and compare that thing with neutral training data using Naïve Bayes technique and returns the calculated neutral probability. Then it returns this value to the compute probability module.
* **Display module**: - It takes the classified data from the main module as the input, and then it displays the classified data on the screen