**Chapter 4 Implementation**

The main objective of our thesis is to detect interrogative Bangla sentences from transliterated Bangla sentences. In general perspective the topic seems very facile but after taking the whole overview, we have discovered that this is a very sophisticated and convoluted problem. We have implemented rule-based approach, supervised learning based approach and deep learning approach.

**4.1 Making of Primary Corpus for Rule-Based Approach**

The dataset we have worked on has been formed by extracting 44,538 comments from cricket -based Facebook public groups. The comments we have extracted contain various types of sentences. We have excluded the comments written in English and in Bangla alphabet and made a dataset and named it as the “Primary Corpus”. From this corpus we have excluded the sentences with the question mark and made the “Interrogative mega Corpus”. From this corpus we have taken the distinct sentences and formed the “Distinct Interrogative Corpus” and the rest of the primary corpus is named as the “Other Mega Corpus”. The “Interrogative Corpus” contains distinct interrogative sentences. We have used these corpora for the rule-based approach of analysis.   
 The information about this corpus is reported in TABLE I.

1. Corpora Analysis

| Corpus | Corpus Information | |
| --- | --- | --- |
| Number of Sentences | Number of Words |
| Primary Corpus | 29,883 | 145,009 |
| Interrogative Mega Corpus | 4,624 | 23,785 |
| Other Mega Corpus | 25,259 | 121,224 |
| Interrogative Corpus | 700 | 3,073 |

**4.2 Editing Corpus for Supervised Learning Based Approach**

With a view to implementing the dataset for supervised learning, we have scrutinized the primary corpora and omitted a large number of sentences which were very raw in nature and had anomalous content. After editing we have got our *Secondary Corpus*. The domain of this corpora is cricket but as it was taken in real time, the corpus contains superb variations.

The information about this corpus is reported in TABLE II.

|  |  |
| --- | --- |
| **Secondary Corpus** | **Number of Sentences** |
| Total Sentences | 8797 |
| Interrogative Sentences | 1704 |
| Non-Interrogative Sentences | 7093 |

After implementing SVM, KNN, MLP and logistic regression classifiers on this corpus we have obtained good results. But we have realized that the result was influenced by the common domain of the training set and test set. Therefore, we tried to add data from another domain as test data. We have data collected for designing a chatbot for admission test. The admission test data was made manually. This dataset follows proper and standard form of transliterated sentences. After adding this dataset with the secondary corpus we have got our *Tertiary Corpus*.

The information about this corpus is reported in TABLE III.

|  |  |
| --- | --- |
| **Tertiary Corpus** | **Number of Sentences** |
| Total Sentences | 11790 |
| Interrogative Sentences | 4138 |
| Non-Interrogative Sentences | 7652 |

**4.3 Implementing Classifiers Based on Supervised Learning**

We have implemented four of the most popular classifiers for dividing our dataset into two classes, interrogative and non-interrogative sentences and calculate the accuracy. We have used scikit-learn for applying SVM, KNN, MLP and logistic regression.

**4.3.1 Labeling of data and Feature Extraction**

We have labeled the interrogative sentences as “1” and the non-interrogative sentences as “0”.

For feature extraction we have a handful of choices. But to keep it simple and automated we have used the CountVectorizer function of the sklearn for finding out the word frequencies. We have set the parameters of the function as follows:

-tokenizer=lambda x: x.split()

- ngram\_range=(1,4)

We keep the other parameters as default.

**4.3.2 Implementing SVM**

We have used the linear kernel SVM of sci-kit learn and the method SVC. We have assigned the parameter *kernel* = “linear”. All the other parameters are kept with their default values. The result we have found by using SVM is quite notable.

**4.3.3 Implementing Logistic Regression**

We have used scikit-learn’s logistic regression model for predicting our classification. We have taken the parameter C= 10 e 15 and keep the other parameters as default. The result of logistic regression for our dataset is somewhat good.

**4.3.4 Implementing MLP**

We have used multilayer perceptron for classification of the interrogative and non-interrogative sentences. We have implied scikit-learn’s MLPClassifier for our purpose. We have assigned the parameters in the following way:

* solver='lbfgs'
* alpha=1e-5
* hidden\_layer\_sizes=(12, )
* random\_state=1

The other parameters are kept with their default values.

**4.3.5 Implementing k-NN**

The k-NN was the least successful classification techniques for our dataset. This happened for the diversity of the dataset and the size of the dataset. We have used scikit-learn’s *k*neighbors classifier. We have assigned the parameter weights='distance' and n-neighbors=3 and kept other parameters with their default values.

**4.4 Deep Learning Implementation**

The result of the above mentioned classifiers are good but we wanted to imply a state-of-the-art approach for the classification. For this, we have used artificial Convolutional Neural Network(CNN) as a deep learning approach. We have used TensorFlow for the CNN. TensorFlow is an open source software library which is used extensively for deep neural network research. The training time was quite less although we have not used GPU. We have used TensorFlow 1.4. We have used our tertiary corpus for CNN. 10% of the data is kept as test dataset. We keep the interrogative and non-interrogative sentences in two separate files. The sentences are preprocessed by adding <PAD> token to make all the sentences of fixed length. The vocabulary is formed by mapping every single words with an integer number. The result we get from applying deep learning is pretty good which is described in Chapter 5.