Sentiment Analysis on Hindi Movie Reviews

A Report Submitted
in Partial Fulfillment of the Requirements
for the Degree of
Bachelor of Technology
in
Computer Science & Engineering

by
Saloni Juneja
Shubham Kumar Goyal
Sonali Agrawal
Saransh Agarwal
Rohit Kumar

to the
COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
MOTILAL NEHRU NATIONAL INSTITUTE OF TECHNOLOGY
ALLAHABAD
November, 2017

UNDERTAKING

We declare that the work presented in this report titled "*Sentiment Analysis on Hindi Movie Reviews*", submitted to the Computer Science and Engineering Department, Motilal Nehru National Institute of Technology, Allahabad, for the award of the ***Bachelor of Technology*** degree in ***Computer Science & Engineering***, is our original work. We have not plagiarized or submitted the same work for the award of any other degree. In case this undertaking is found incorrect, we accept that our degree may be unconditionally withdrawn.

November, 2017
Allahabad

Saloni Juneja(20144057)
Shubham Kr. Goyal(20145068)
Sonali Agrawal(20144017)
Saransh Agarwal(20144119)
Rohit Kumar(20144110)

CERTIFICATE

Certified that the work contained in the report titled "*Sentiment Analysis on Hindi Movie Reviews*", by Saloni Juneja, Shubham Kumar Goyal, Sonali Agrawal, Saransh Agarwal and Rohit Kumar has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

(Dr. Rupesh Kumar Dewang
)
Computer Science and Engineering Dept.
M.N.N.I.T, Allahabad

November, 2017

Preface

It is human behaviour to look for others opinion before taking any decision. A lot of documents are available which express opinions on different issues. But the main challenge arises in analyzing these documents to produce useful knowledge. Tremendous works in the area of Sentiment Analysis is available for English language. However, there has been little work done for Indian languages. From the last few years, opinion-rich resources are booming in Hindi and hence there is a need to perform Sentiment Analysis in Hindi.

In this report, we have categorized movie reviews in Hindi as positive or negative. Two methods- Unigrams and TF-IDF have been used for feature matrix generation. Then we have applied Deep Belief Network for classification and compared results with other classifiers. Others approaches have also been implemented and results obtained have been compared.

Acknowledgement

We take this opportunity to express our profound gratitude and deep regards to The Director, Dr. Rajeev Tripathi and The Head of our Department, Prof. Neeraj Tyagi. We also convey our regards to The DUGC Convener, Dr. Shashank Srivastava for giving us opportunity to work on such interesting projects. We would heartily like to thank our guide Dr.Rupesh Kumar Dewang, Department of Computer Science and Engineering for his exemplary guidance, monitoring and constant encouragement throughout the course of this project.

We are thankful to and fortunate enough to get constant encouragement and support from all teaching staffs of Department of Computer Science which helped us in successfully completing our project. Also we would like to extend our sincere regards to all the non-teaching staff of Department of Computer Science for their timely support.

Contents

Chapter 1

Introduction

Sentiment Analysis is a natural language processing task which helps to identify and categorize opinions expressed in a piece of text as positive, negative or neutral [5]. It helps to determine the reviewer's point of view on a particular topic. It combines the techniques of computational linguistics and Information Retrieval (IR). The increasing user-generated content on the Internet is the motivation behind the sentiment analysis research.

Hindi is fourth highest speaking language in the world. It is spoken nearly by 425 million people as first language and 120 million people as a second language. Lots of websites, blogs and tweets now a days support Hindi language and some of them use Hindi as a primary language as well. But most of the research on Sentiment Analysis has been focused on English language and very less attention is paid in direction of sentiment analysis in Hindi language. Increasing user-generated content in Hindi on the internet has motivated us to perform sentiment analysis research on movie reviews in Hindi.

Sentiment Analysis in Hindi is very challenging due to the following reasons:

- Hindi is a resource scarce language which causes problems in collection and generation of datasets. Also, there are not efficient parsers and taggers for this language.

- Hindi is morphologically rich and a free order language as compared to English language. It means there is no specific arrangement of words in Hindi language

i.e. subject, object and verb comes in any order whereas English is fixed word order language i.e. subject is always followed by a verb and then followed by an object. Word order is important for determining the polarity of a given text.

- Unavailability of well annotated standard corpora.
- Limited resources are available for it like Hindi SentiWordNet (H-SWN). It consists of limited numbers of adjectives and adverbs.

Application of Sentiment Analysis are endless.

- Sentiment analysis has been used by e-commerce companies for customer satisfaction. You can estimate how happy customers are, by estimating the ratio b/w positive and negative reviews.
- Sentiment Analysis is used everyday in social media, surveys, feedbacks to identify the needs of the people.
- To identify the detractors and promoters.

In our project, we perform Sentiment Analysis on movie reviews in Hindi. Sentiment Analysis is a text classification problem in which a text is assigned as positive, negative or neutral depending on the sentiment which it strongly forces. We have used various methods to perform classification which can be found in Chapter 3.

Chapter 2

Related Work

Very few research work has been done related to sentiment analysis in Hindi. The earliest of them was by Aditya Joshi et al [11]. They proposed that a fall-back strategy could be adopted for doing sentiment analysis for a new language. They suggested that we could first of all train a sentiment classifier on in-language labeled corpus and use it to classify a new document. In case of in-language data not being available, rough machine translation could be applied to convert the document into a resource rich language like English. Hence, the polarity of translated document could be predicted using a classifier for English, assuming polarity is not lost in translation.

An important contribution to Hindi Polarity Classification was done by Bakliwal et al [9]. Their major contributiob was that they created a resource for Hindi by using Hindi WordNet to retrieve synonyms and antonyms of a given word in Hindi for which they knew the polarity and then assigned the similar polarity to synonyms and opposite polarity to antonyms. They also developed an annoted corpora of Hindi Product Reviews.

An efficient approach was developed by Namita mittal et al. [12] based on negation and discourse relation for identifying the sentiments from Hindi content. The annotated corpus for Hindi language was developed and existing Hindi SentiWordNet (H-SWN) was improved by incorporating more opinion words into it. They also devised the rules for handling negation and discourse that affect the sentiments

expressed in the review.

Another important work was done by Piyush Arora [8]. He proposed a technique to build a subjective lexicon given a pre-annotated seed list for a language and its WordNet representing the network/connectivity of words using synonyms and antonyms relations. One of the salient features of his approach is that his technique could be applied for any language which has the WordNet available.

Namam Bansal et al [10] proposed a semi-supervised approach to train a Deep Belief Network on a small percentage of labeled data and assign polarity to unlabeled data. They used semi-supervised learning because supervised polarity classification systems are domain-specific and hence systems trained on one dataset typically perform much worse on a different dataset. They also stated that annotating a large amount of data could be an expensive process.

A novel approach was proposed by Richa Sharma et al. [14] in which they developed a Hindi language opinion mining system to classify reviews as positive, negative or neutral. They also handled negation in their proposed system. Instead of using Wordnet, they developed their own Hindi dictionary to determine the polarity of Hindi reviews.

In our project, we have analyzed and implemented two research papers namely, Sentiment Analysis in Hindi by Naman Bansal et. al and Sentiment Analysis for Hindi Language by Piyush Arora. We have compared accuracy and f-measure obtained by following the models implemented in above mentioned papers.

Chapter 3

Work Done

In the section, we have proposed the details of dataset and models used for reviews preprocessing, algorithms used for feature set generation and various classifiers used.

3.1 Dataset Used

We have used 250 sentences (125 positive and 125 negative) of movie reviews available from IIT Bombay for research purposes [10]. In addition to this, we have manually collected and labeled around 750 sentences of movie reviews (375 positive and 375 negative reviews) from Hindi review site (jagran.com). In total, we have a dataset of 1000 movie reviews.

3.2 Data Preprocessing

We have removed all those words from each review which do not contribute to the accuracy of classification. These words are basically as follows:

- **Punctuations:** They are symbols such as ",","—","!" etc. which are used in writing to separate sentences and their elements and to clarify the meaning.
- **Numbers:** Numbers do not contribute to the accuracy and has no meaning in sentiment analysis. So they are removed during preprocessing.

- **Stop words:** The natural language words which have very little meaning such as articles, pronouns and prepositions are denoted as stop words.
- One length words, etc.

3.3 Feature Extraction

Once the preprocessing of data is done, we compute the feature matrix using the TF-IDF and unigram model.

3.3.1 TF-IDF algorithm

In our code, we have used TfidfVectorizer() function available in scikit-learn library [13]. It is used to convert a collection of raw documents to a matrix of TF-IDF features. The goal of using TF-IDF instead of the raw frequencies of occurrence of a token in a given document is to scale down the impact of tokens that occur very frequently in a given corpus and that are hence empirically less informative than features that occur in a small fraction of the training corpus.

3.3.2 Unigram model

Unigram model considers each word at a time. It doesn't take word ordering into account, so the order doesn't make a difference in how words are tagged or split up. In this model, we create a lexicon containing all the words that occur in any review of our dataset. Lexicons are the set of combined word of all the positive and negative reviews. We consider only those words in the lexicon which have frequency count in a specific range in order to eliminate those words that do not contribute much in sentiment classification. We generate a feature matrix of size m*n (where m= number of reviews in our dataset and n= number of words in the lexicon). For each element of the matrix, if that lexicon word occurs in the review, the element is assigned frequency count of that word in the review.

Algorithm 1 Feature matrix generation using unigram model

Input:

We have a list of reviews, **R** which contain positive reviews and negative reviews in string format. $\mathbf{R} = \{p_1, p_2....p_{m_1}, n_1, n_2....n_{m_2}\}$, where m_1 = number of positive reviews and m_2 = number of negative reviews

Output:

A feature set **F**, which is list of features for each review. **F**= $\{f_1, f_2...f_{m_1}, f_{m_1+1}...f_{m_1+m_2}\}$
size of feature set **F** : (m1+m2)*n
where n = the number of features in a single review that is equal to the length of lexicons set

Create a set of lexicons **L**.

For each review r_i in **R**:

tokenized_words = word_tokenize(r_i)

words = preprocessing(tokenized_words)

L+=words

ENDFOR

For each review r_i in **R**:

F = list along with features and labels

features = list of zeros, size equal to length of lexicons set **L**

For each word w in r_i

If w exist in **L**

index = **L**.index(w)

features[index] += 1

If it belongs to positive review

F.append([features,1])

Else

F.append([features,0])

ENDFOR

Shuffle the Feature set **F**

3.4 Approaches Used

The preprocessed movie reviews are classified using three methods:

- Resource based classification using HindiSentiWordNet(H-SWN)
- In-language classification through various classifiers like Deep Belief Network, Neural Network, SVM etc.
- Machine Translation- Based Semantic Analysis

3.4.1 Resource Based Classification

A simple approach to predict the sentiment of a review is to use the prior polarity of terms present in it. In order to find the polarity, a lexical resource is required. In this method of classification, we use Hindi SentiWordNet(H-SWN) as the resource for developing majority based sentiment classifier.

Each word present in the H-SWN has a positive and a negative sentiment score. Based on the maximum of the scores, a polarity is assigned to each word in a review. The polarity which covers the maximum number of words in a review is predicted as the sentiment of that review.

Algorithm 2 Resource based classification using Hindi SentiWordNet

Input:

1) A list of reviews, **R** which contain positive reviews and negative reviews in string format. $\mathbf{R} = \{p_1, p_2....p_{m_1}, n_1, n_2....n_{m_2}\}$, where m_1 = number of positive reviews and m_2 = number of negative reviews

2) Hindi SentiWordNet dictionary **dict** which contain a tuple for every word having its positive polarity score and negative polarity score.

Output:

A list **P** containing the polarity of reviews

Make a list of polarity of reviews **P**=[]

For each review r_i in **R**

    Apply preprocessing on r_i

    Make a list of votes v=[]

    Initialize two variables x1=0, x2=0

    For each word w in r_i

        if w exists in **dict**

            pos_score, neg_score = dict[w]

            if pos_score > neg_score

                v.append(1)

                x1+=pos_score

            else

                v.append(0)

                x2+=neg_score

            else

ignore the word

ENDFOR

x = number of ones in the list

y = number of zeros in list

if x > y

sense = 1 (here 1 denotes positive)

else if y > x

sense = 0 (here 0 denotes negative)

else

if x1 > x2

sense = 1

else

sense = 0

P.append(sense)

ENDFOR

3.4.2 In-language Classification

This approach is based on training the classifiers on the same language as the text. It relies heavily on availability of resources in the same language to analyze the sentiment. Thus all training text, testing text are in Hindi language. The feature representation(Term frequency or TF-IDF) can be varied to see the effect on In-language classification on Hindi reviews. In this approach, we use a variety of classifiers to train and test the data. We know TF-IDF can be a better way of feature matrix generation as it reduces effect of very frequent words in document but do not contribute much to the relevance of text.

3.4.3 Machine Translation Based Semantic analysis

There is scarcity of resources in Hindi, that enforces us to take into consideration the machine translation based semantic analysis approach [11]. The idea behind machine translation based semantic analysis is to model a classifier on standard English movie reviews. Then a translation module(here, Google Translate) is used to translate the reviews in Hindi to English. The model can then be used to classify the translated documents. Here, the result is reported only for TF-IDF representation of feature matrix run on Decision Tree classifier.

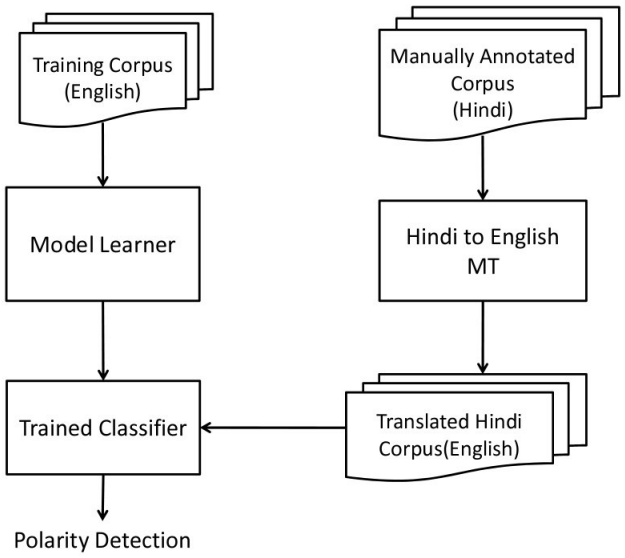

Figure 1: Procedure for MT based Sentiment Analysis

3.5 Classification

This section contains different classification models that we have used to predict the polarity of movie reviews in Hindi.

3.5.1 Deep Neural Network

Deep Neural Network [2] consists of systems of interconnected "neurons" capable of computing or processing values from inputs. They are designed to recognize patterns. The patterns they recognize are numerical, contained in vectors, into which all real-world data, be it images, sound, text or time series, must be translated. The network is formed by connecting the output of certain neurons to the input of other neurons. This forms a directed and weighted graph, where the neurons are the nodes and the connection between the neurons are weighted directed edges.

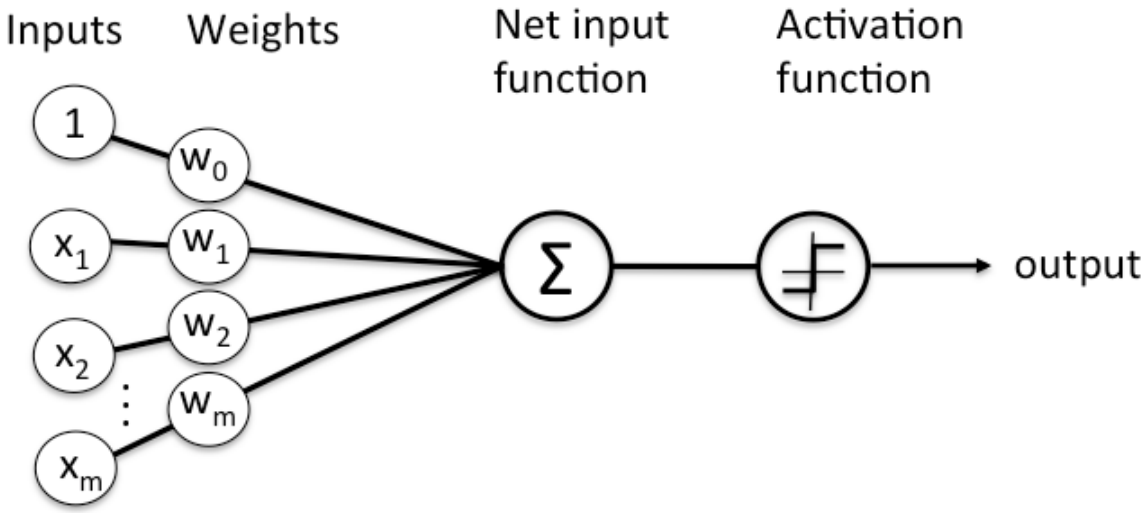

Figure 2: How a single node looks like in a neural network

3.5.2 Naive Bayes

Naive Bayes is a classification technique based on Bayes' Theorem with an assumption of independence among predictors [3]. It assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. Naive Bayes based classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem.

3.5.3 Deep Belief Network

Deep Belief Networks are similar to neural networks but they differ in the number of hidden layers. Deep Belief Networks have multiple hidden layers stacked one over the other to capture the complex non-linearity in the data. They essentially disentangle the underlying variation in the data.

DBNs can be viewed as a composition of simple, unsupervised networks such as restricted Boltzmann machines (RBMs) [4] where each sub-network's hidden layer serves as the visible layer for the next. An RBM is an undirected, generative energy-based model with a "visible" input layer and a hidden layer and connections between but not within layers.

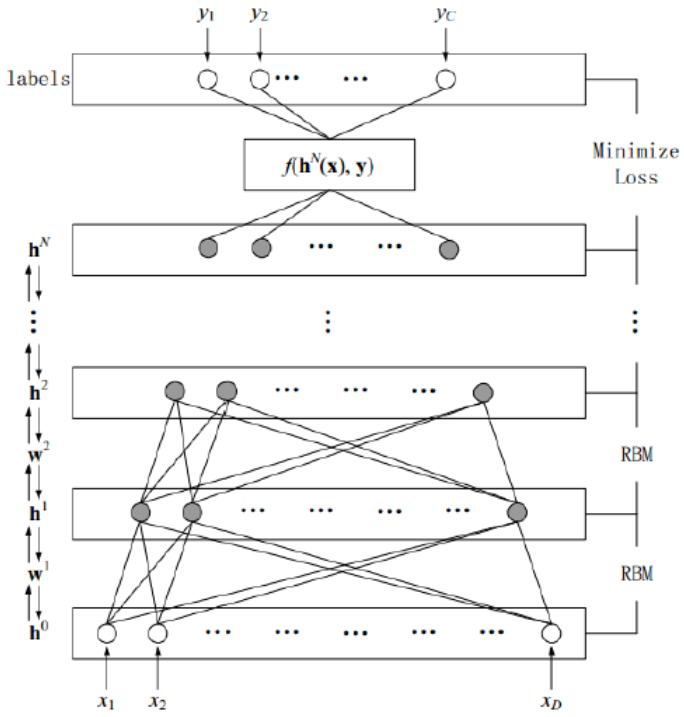

Figure 3: Architecture of Deep Belief Network

3.5.4 Logistic Regression

Logistic regression measures the relationship between the categorical dependent variable and one or more independent variables by estimating probabilities using a logistic function, which is the cumulative logistic distribution. Logistic regression is a linear method, but the predictions are transformed using the logistic function. It is a statistical method for analyzing a dataset in which there are one or more independent variables that determine an outcome.

3.5.5 Support Vector Machine

Support Vector Machine(also known as Support Vector Networks) [6] is a supervised learning model which contains various learning algorithms to analyze classification and regression problems. It is also known as binary classifier which attempts to find a hyperplane that can separate two class of data by the largest margin. Given a set of points of two types in N- dimensional place, SVM generates a (N-1) dimensional hyperplane to separate those points into two groups. SVMs can be used to solve various real life problems like classification of images, recognizing hand written characters, categorizing text and hypertext, etc.

3.5.6 Decision Tree

Decision Trees (DTs) [1] are a non-parametric supervised learning method used for classification and regression. A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute and each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes). The paths from root to leaf represent classification rules. The goal is to create a model that predicts the value of a target variable by learning simple classification rules inferred from the data features.

3.5.7 Voting Classifier

Voting classifier is a kind of Ensemble method. The goal of Ensemble methods is to combine the predictions of several base estimators with a given learning algorithm

in order to improve the accuracy and robustness of the classifier. [7] The idea behind the Voting Classifier is to combine conceptually different machine learning classifiers and use a majority vote or the average predicted probabilities (soft vote) to predict the class labels. Such a classifier can be useful for a set of equally well performing model in order to balance out their individual weaknesses.

Chapter 4

Experimental Setup and Results Analysis

In our project, we have used the dataset of 1000 movie reviews in Hindi. They are manually labeled into two classes- positive and negative. Then we have generated a feature matrix using TF-IDF and unigram models. The obtained feature matrix is fed into different classifiers. We have used following classifier in our project: Naive Bayes, Logistic Regression, Support Vector Machine, Decision Trees, Neural Network and Deep Belief Network.

Apart from predicting classes using feature matrix, we have also used Resource based Classification Technique using Hindi SentiWordNet(H-SWN). We have also attempted Machine Translation based Sentiment Analysis by first converting the reviews in Hindi to English using Google Translator. In this model, the classifier(Decision Tree) is trained on a large database of movie reviews in English. Thus, our dataset is entirely used for testing.

All the results are reported for 10-fold cross validation. But in case of deep belief network and deep neural network,we have used 80-20 percent split. We just calculated accuracy by comparing actual class label of reviews against the predicted ones. We also compare accuracy obtained using various classifiers.

Actual Value

Predicted Value		positives	negatives
	positives	True Positive(t_p)	False Positive(f_p)
	negatives	False Positive(f_p)	False Negative(f_n)

Table 1: Confusion Matrix

We consider the following evaluation measures in order to compute the overall performance of the system.

1. **Precision:** Precision is defined as portion of true positive predicted instances among all positive predicted instances.

$$Precision = \frac{t_p}{t_p + f_p}$$

2. **Recall:** Recall is calculated as portion of true positive predicted instances against all actual positive instances.

$$Recall = \frac{t_p}{t_p + f_n}$$

3. **Accuracy:** Accuracy basically is the portion of true predicted instances against all predicted instances.

$$Accuracy = \frac{t_p + t_n}{t_p + t_n + f_p + f_n}$$

4. **F-measure:** F-measure is the combination of Presicion and Recall and is calculated as:

$$F - measure = \frac{2 * Precision * Recall}{Recall + Precision}$$

Classifiers used	**Unigram(%)**	**TF-IDF(%)**
Resource based classifier	53.51	53.51
Logistic Regression	78.98	85.24
Stochastic Gradient Descent	75.46	90.05
MultiNomial Naive Bayes	77.8	85.14
Support Vector Machine	50.4	85.24
Decision Tree	72.08	90.85
Voting Classifier	79.09	89.94
Neural Network	61.05	70.99
Deep Belief Network	50.5	54.5
Decision Tree Classifier(in case of translation)	54.5	72.5

Table 2: Accuracy obtained using different classifiers

Classifiers used	**Unigram**	**TF-IDF**
Resource based classifier	0.527	0.527
Logistic Regression	0.7853	0.9303
Stochastic Gradient Descent	0.7661	0.9433
MultiNomial Naive Bayes	0.7848	0.9298
Support Vector Machine	0.6702	0.9278
Decision Tree	0.72	0.9533
Voting Classifier	0.7628	0.9491
Deep Belief Network	0.58	0.67
Neural Network	0.58	0.68
Decision Tree Classifier(in case of translation)	0.654	0.8382

Table 3: F-measure obtained using different classifiers

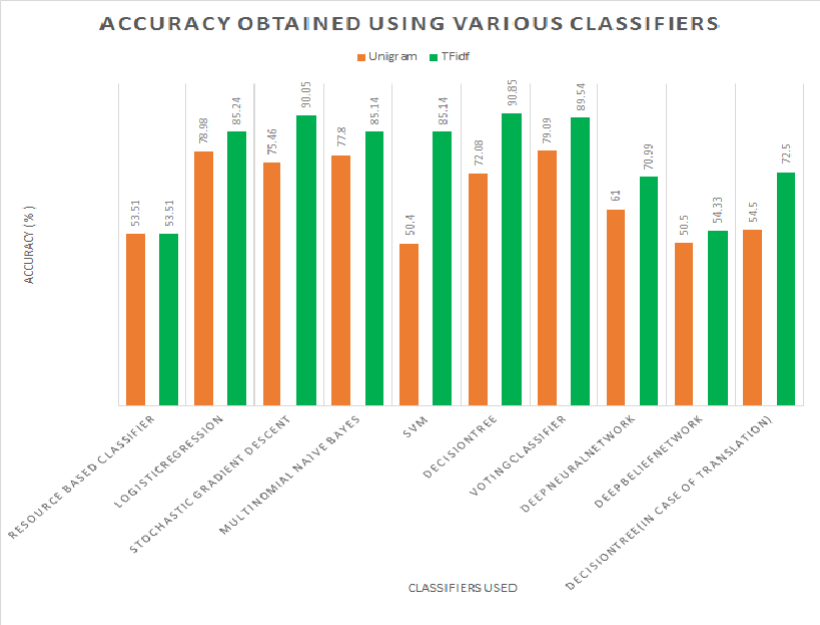

Figure 4: Accuracy Graph for different classifier algorithms with different models

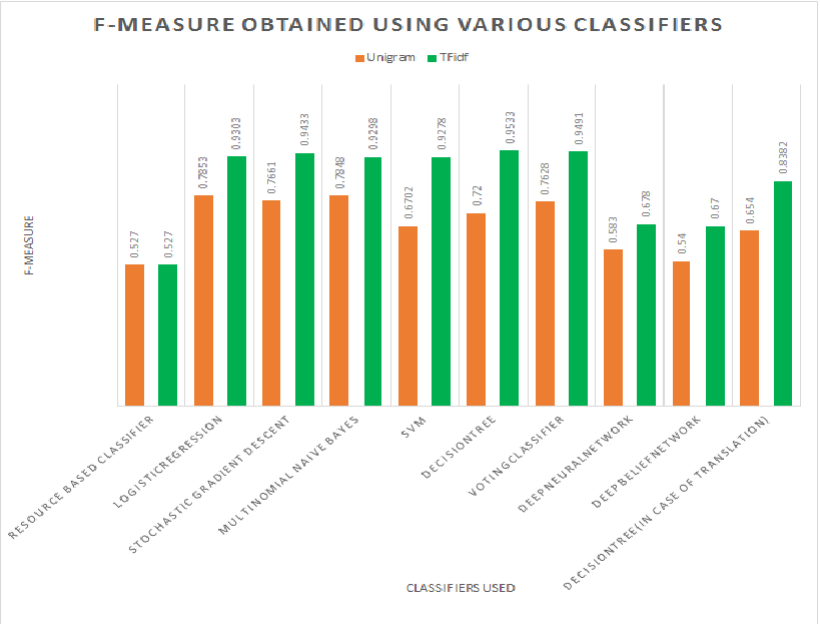

Figure 5: F-measure graph for different approaches with different models

Chapter 5

Conclusion and Future Work

In our project, we have mainly focused on three approaches. The first approach involved using a majority-based classifier for Hindi SentiWordNet. The second approach focused to train a model on annotated English corpus and translate a Hindi document to English in order to use this model. In final approach we constructed a classifier model for Hindi using a training corpus in Hindi. The result proved that the third approach is better among the others. This means that best result can be achieved with an annotated corpus in the same language of analysis. Also, among the two models- unigram and TF-IDF used in the third approach, TF-IDF proves to generate a better result than the other. MT-based systems give superior classification performance as compared to majority-based systems based on lexical resources.

In future, we can extend resource-based sentiment analysis to include Word Sense Disambiguation(WSD) so that a specific sense of word can be looked up in the H-SWN. Since Hindi SentiWordNet covers only limited number of words at present, we can extend our work to cover more number of words by improving H-SWN. This will help us in achieving better accuracy. Further we can expand our approach to handle negation rules which is not supported by our present models.

References

[1] Decision Tree. https://en.wikipedia.org/wiki/Decision$_t ree$.

[2] Deep Neural Network. https://deeplearning4j.org/neuralnet-overviewdefine.

[3] Naive Bayes. https://en.wikipedia.org/wiki/Naive$_B ayes_c lassifier$.

[4] RBM wikipedia. https://en.wikipedia.org/wiki/Restricted$_B oltzmann_m achine$.

[5] Sentiment Analysis wikipedia. https://en.wikipedia.org/wiki/Sentiment$_a nalysis$.

[6] Support Vector Machine. https://en.wikipedia.org/wiki/Support$_v ector_m achine$.

[7] Voting Classifier. https://en.wikipedia.org/wiki/Ensemble$_l earning$.

[8] Arora, P. Sentiment analysis for hindi language. *MS by Research in Computer Science* (2013).

[9] Bakliwal, A., Arora, P., and Varma, V. Hindi subjective lexicon: A lexical resource for hindi polarity classification. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC)* (2012), pp. 1189–1196.

[10] Bansal, N., Ahmed, U. Z., and Mukherjee, A. Sentiment analysis in hindi. *Department of Computer Science and Engineering, Indian Institute of Technology, Kanpur, India* (2013), 1–10.

[11] Joshi, A., Balamurali, A., and Bhattacharyya, P. A fall-back strategy for sentiment analysis in hindi: a case study. *Proceedings of the 8th ICON* (2010).

[12] Mittal, N., Agarwal, B., Chouhan, G., Pareek, P., and Bania, N. Discourse based sentiment analysis for hindi reviews. In *International Conference on Pattern Recognition and Machine Intelligence* (2013), Springer, pp. 720–725.

[13] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research 12* (2011), 2825–2830.

[14] Sharma, R., Nigam, S., and Jain, R. Polarity detection movie reviews in hindi language. *arXiv preprint arXiv:1409.3942* (2014).