Opinion Mining on Hindi-English Code Mixed Data

Abstract

Opinion Mining which also goes by the name Sentiment Analysis is one of the hot research areas. Here, we focus on problem of sentiment analysis of Hindi-English code mixed data. One can observe code-mixing in user generated content on social media, especially from multilingual users (people knowing more than one language). Such data neither has any specific spelling standards nor any formal grammar rules. We approached by first crawling raw data from Facebook pages of celebrities, politicians, newspapers and some other comic content, and then manually refining & labeling the data. Finally we applied a set of NLP and machine learning methods to solve this task at hand. To the best of our knowledge, we are the pioneers to this type of work.

1 Introduction

Opinion Mining is about identifying the hidden sentiment of people about a certain topic under discussion. This problem has been addressed for one language datasets namely Hindi [*Joshi et al, 2010*], Spanish, English [*Agarwal et al, 2011*], Bengali etc. This task itself is challenging, but it becomes even more interesting when we have data with two or more languages used simultaneously. This sort of data where more than one language is used interchangeably is classified as code-mixed or code-switched. Although some authors in literature use both code-mixing and code-switching interchangeably but yet there is a slight difference which should be addressed when having task like opinion mining at hand. Code mixing is referred as using two or more languages in speech whereas code switching occurs when speaker alternates between languages.
For instance, *Yeh College bahut accha hai* is an example of Hindi-English code mixing and *mein wahan gaya tha but the shop was closed!!!* is an example of code-switching. In code-mixing, just one or two words of other language is used whereas in code-switching the sentence could be broken into two parts and it preserves the grammar rules of that particular language in those sub-parts. In code mixing, one language is a matrix language while the other is subordinate. And we have in most cases that the native language or the mother tongue is matrix language. Since here we are addressing code mixing of Hindi-English code mixed speech samples, the matrix language will be either Hindi or English and subordinate language will be the other one. So, this characteristic of code-mixing will be helpful in feature engineering phase.
To the best of our knowledge, this is the first attempt at opinion mining for Hindi-English code-mixed data or code-mixed data in general.

2 Dataset Description

2.1 Corpus Creation

For this exploration on code mixed Hindi-English opinion mining, we collected raw data from some very popular Facebook public pages like *Narendra Modi*, *Aam Aadmi Party*, *Garbage Bin*, *BBC Hindi*, *ABP News*, *Doordarshan*, *Dainik Jaagran* and *Dainik Bhaskar*. These pages cover themes like politics, comic, fan following. These pages have mostly code-mixed posts and comments which gives a sense of debate/discussion on social media. A total of around 25,752 comments were collected of which we manually selected and labeled 7663 comments, into three categories namely: positive, negative and neutral.

2.2 Data Cleaning

The data so obtained was noisy and followed no case rules. By analyzing the data we observed that users post URLs in their comments. We normalized all URLs and replaced them with some common string "HTTPURL." Data was converted entirely to lower case so as to match with other resources data format. For punctuation marks "." , ",", "!" and "?"we inserted single whitespace before and after these punctuations. This was done to obtain the exact word which was written before and after the punctuation.

3 Resources and Preprocessing

3.1 Dictionaries:

For prior polarity based features we used following resources. (a) Hindi Sentiwordnet by IITB (b) Hindi Lexicon scores dictionary by IIITH (c) English Sentiwordnet (d) Ubuntu system English dictionary. By using resource (a) we created a new dictionary by first converting words from Devnagri to WX notation by using IIITH-Shallow parser tool. Then from WX notation we converted words to roman scripts using set of *WX-to-roman* conversion rules and normalized (*discussed in next subsection*) them to finally put them in our dictionary. For e.g. see below table.

Original word (in Devnagri)	**Intermediate word (WX Notation)**	**Words placed in our dictionary**
अच्छा	acCA	acchaa, accha, acha
गुलाम	gZulAma	gulaama, gulama, gulaam, gulam

Similarly, we built another dictionary from resource (b) using the same above technique. The score were similar to the one from the original.

3.2 Normalization:

The non-formalism of Hindi-English code mixed data is a major hindrance. Since every person has his/her own way of writing Hindi words in Roman script, we need all or at least maximum possible combinations of words which could be written by people. For *e.g.*, word ***अच्छा*** could be generally written as *acchaa*, *accha* or *acha*. So we used a simple yet elegant normalizing technique on Hindi words in Roman script. Then we make all the variations of the word and the word obtained by replacing all repeated adjacent characters by only one character i.e. by itself, by taking all combinations of vowels between the first and the last characters.

3.3 Negation Handler:

Handing negation is very important to get the correct opinion of the sentence. In code mixed data negation handling is also a bigger challenge. As we know, there is a fixed order of subject-verb-object (SVO) in English language but in Hindi language there is no fixed order means order can be SVO or SOV. Because of this if there is any Hindi negation word like ***nahi*** in the sentence we replaced all the words before it up to previous punctuation mark if there exists and all words after it up to next punctuation mark, by neg_word (word in normalized form if the word is of Hindi) and if we encounter the English negation word like ***not*** then we replace all words only after that negation word by neg_word up to the punctuation mark if exist otherwise to the end of the sentence.

4 PMI

This technique is used to generate a new resource which signifies the weight of the label of a word in the training set as positive, negative and neutral. For each word, this newly created resource contains the weight value for each of the possible classes namely positive, negative and neutral. This resource can be used as a dictionary in which we can look up words from test dataset for prediction of weight values on one of the three classes.

This dictionary maintains counts of each word from training dataset for each category. The counts resemble how many times the given word occurs in respectively labeled category sentence. For example:

Comment	*Label*
Bahut acha comic hai ye …	1
Wo restaurant acha nahi hai !	-1
Class khatam ho jaati tab aata hai guddu !	0

Word	Positive	Negative	Neutral
Acha	1	1	0
Khatm	0	0	1
Bahut	1	0	0

In above table, the word "*acha*" occur once in a positive labeled sentence, once in negative labeled sentence and does not occur in neutral sentence.

5 Features

5.1 Word-n-grams:

Unigrams (*f1*) and bigrams (*f2*) are considered for generating the feature vector. Prior to generation of unigrams and bigrams, negation-handling and normalization is applied on the data and the modified train data file is considered for n-grams generation. Stop-words from a standard stop-

word-list are also removed in this step. N-grams contribute to the major portion of the features array which we are using for input to *support vector machine*.

5.2 Lexicon Score:

We used the four features for English and Hindi separately. These features are *total positive score*, *total negative score*, *total positive count*, *total negative count*, *max positive* and *max negative (all together comprise of f3)*. In a data sentence for every word we look up the word in English Sentiwordnet dictionary. If it contains the word then we take the sentiment score of that word and added to corresponding feature namely positive or negative. We also maintain the max positive and max negative scores. If word was not found in English Sentiwordnet dictionary then we look up the word in two Roman script based dictionaries for Hindi words that we created in which all combinations of Hindi words are present with their sentiment scores and we obtained the above four features for Hindi words also.

5.3 Emoticons:

Emoticons (*f4*) play an important role in identifying the sentiment of the sentence on social media. Majority of the users use smileys while writing comments, posts, etc. on social media. So, this feature plays an important role in detection of the sentiment of the sentence. In our project, we used 4 different categories of smileys as types of features viz.

Class	*Emoticon*
Positive	:-), :), :D
Negative	:-(, :'(
Playful	;), :P
Love	:-*, :-{}

5.4 POS tagging:

As the code mixed data does not have any generalized grammatical rules for any of the two languages, we considered using PoS tagger for both the languages. We are only considering non-polar PoS tags on unigrams for our purpose. For English language, we used ARK PoS tagger which was originally written for twitter data. We extracted around *27* PoS tags defined by tagger. We simply maintained counts of the PoS tags for a particular sentence as part of our feature vector. For PoS tagging in Hindi, we used Shallow Parser. They also contributed to *25* another features by simply adding counts of the PoS tags. This comprised of our feature *f5*.

5.5 PMI based scoring:

We used scaled PMI scores, *f6*, from PMI dictionary as discussed in *section 4*. Scaling PMI scores is done with respect to the total score for that word in each of the three categories. PMI based scoring is mainly useful where a generalized dictionary is not available. The cumulative sum of PMI scores for each word of the sentence is scaled down by total scores for each of the category for every word of that sentence and this value is used as a feature in feature vector. The features generated by this method are "Positive PMI Score, Negative PMI score and Neutral PMI score". This method is fully based on the training data and training data decides the polarity of the word.

6 Experiments and Feature Analysis

The table shown below denotes the experimental results for different combinations of features. For each combination, accuracy using libSVM is shown. Abbreviations used are:

f1 – Unigram
f2 – Bigram
f3 – Lexicon Score
f4 – Emoticons
f5 – PoS Tags
f6 – PMI Scores

Features Used	***Accuracy (in %)***
f1	55.73
f1+f2	58.66
f3	54.93
f1+f2+f3	59.20
f1+f2+f3+f4	59.13
f1+f2+f4	58.60
f1+f2+f4+f5	58.93
f1+f2+f3+f4+f5+f6	60.73
f1+f2+f3+f4 +f6	60.60
f1+f2+f4+f6	59.80
f1+f2+f6	58.53

7 Conclusion

Our paper mainly discussed the problem of determining sentiment in Hindi-English code mixed data. We used unigrams and bigrams as our baseline and added four more features for handling the sentiments in a more accurate way.

As very little work has been done for code mixing, we had to start from scratch and build our own resources.

References

Agarwal, Apoorv et al. "**Sentiment analysis of twitter data**." *Proceedings of the Workshop on Languages in Social Media* 23 Jun. 2011: 30-38.

Joshi, Aditya, AR Balamurali, and Pushpak Bhattacharyya. "A fall-back strategy for sentiment analysis in hindi: a case study." *Proceedings of the 8th ICON*(2010).

Kachru, Braj B. "**Toward structuring code-mixing: an Indian perspective**."*International Journal of the Sociology of Language* 1978.16 (1978): 27-46.

Gupta, Kanika, Monojit Choudhury, and Kalika Bali. "Mining Hindi-English Transliteration Pairs from Online Hindi Lyrics." *LREC* 2012: 2459-2465.

Gupta, Kanika, Monojit Choudhury, and Kalika Bali. "Mining Hindi-English Transliteration Pairs from Online Hindi Lyrics." *LREC* 2012: 2459-2465.

Vyas, Yogarshi et al. "**Pos tagging of english-hindi code-mixed social media content**." *Proceedings of the First Workshop on Codeswitching, EMNLP*2014.