A Dataset for Detecting Irony in Hindi-English Code-Mixed Social Media Text

by

Deepanshu Vijay, Aditya Bohra, Vinay Singh, Syed S. Akhtar, Manish Shrivastava

in

15th Extended Semantic Web Conference
(*ESWC-2018*)

Greece

Report No: IIIT/TR/2018/-1

Centre for Language Technologies Research Centre
International Institute of Information Technology
Hyderabad - 500 032, INDIA
June 2018

A Dataset for Detecting Irony in Hindi-English Code-Mixed Social Media Text

Deepanshu Vijay*, Aditya Bohra*, Vinay Singh, Syed S. Akthar, and Manish Shrivastava

Language Technology Research Centre, International Institute of Information Technology, Hyderabad,
{deepanshu.vijay,aditya.bohra,vinay.singh, syed.akhtar}@research.iiit.ac.in
m.shrivastava@iiit.ac.in

Abstract. Irony is one of many forms of figurative languages. Irony detection is crucial for Natural Language Processing (NLP) tasks like sentiment analysis and opinion mining. From cognitive point of view, it is a challenge to study how human use irony as a communication tool. While relevant research has been done independently on code-mixed social media texts and irony detection, our work is the first attempt in detecting irony in Hindi-English code-mixed social media text. In this paper, we study the problem of automatic irony detection as a classification problem and present a Hindi-English code-mixed dataset consisting of tweets posted online on Twitter. The tweets are annotated with the language at word level and the class they belong to (Ironic or Non-Ironic). We also propose a supervised classification system for detecting irony in the text using various character level, word level, and structural features.

Keywords: code-mixing, language detection, linguistics, svm, random forest, hate-speech.

1 Introduction

Irony is a subtle form of humor, where there is a gap between the intended meaning and the literal meaning. Even though it is a widely studied linguistic phenomenon, no clear definition seems to exist [5]. Irony detection is a difficult task as irony often has ambiguous interpretations. Apart from it's importance in sentiment analysis and opinion mining, irony detection is also vital in the areas of medical care and security [6]. Previous research related to this task has mainly been focused on monolingual texts [18, 2, 8, 5] due to the availability of large-scale monolingual resources. Popularity of opinion-rich online resources like review forums and microblogging sites has encouraged users to express and convey their thoughts all across the world in real time. In multilingual societies like India, users often interchange between two or more languages while communicating online.

* These authors contributed equally to this work.

Code-Mixing (CM) is a natural phenomenon of embedding linguistic units such as phrases, words or morphemes of one language into an utterance of another [13–15, 4]. English and Hindi are two of the most widely used languages in the world and to the best of our knowledge currently there are no online Hindi-English code-mixed resources available for detecting irony.

Following are some instances of Hindi-English code-mixed tweets. It can be observed that **T1** and **T2** contain irony while **T3** is a non-ironic tweet.

T1 : *"Wo ek teacher hai tab bhi life ke test mein fail ho gaya! Hahaha such irony :D"*
Translation : "He is a teacher yet he failed in the test of life! Hahaha such irony :D."

T2 : *"The kahawat 'old is gold' purani hogaee. Aaj kal ki nasal kehti hai 'gold is old', but the old kahawat only makes sense. #MindF #Irony."*
Translation : "The saying 'old is gold' is old. Today's generation thinks 'gold is old' but only the old one makes sense. #MindF #Irony. "

T3 : *"mere single hone ke bawzood mujhe ye nahi pata tha aaj rose day he #irony."*
Translation : "Inspite of me being single, I didn't know today is rose day #irony."

The structure of the paper is as follows. In Section 2, we review related research in the area of code mixing and irony detection. In Section 3, we describe the corpus creation and annotation scheme. In Section 4, we present our system architecture which includes the pre-processing steps and classification features. In Section 5, we present the results of experiments conducted using various character-level, word-level and structural features. In the last Section, we conclude our paper, followed by future work and references.

2 Background and Related Work

[11] performed analysis of data from Facebook posts generated by English-Hindi bilingual users. Analysis depicted that significant amount of code-mixing was present in the posts. [21] formalized the problem, created a POS tag annotated Hindi-English code-mixed corpus and reported the challenges and problems in the Hindi-English code-mixed text. They also performed experiments on language identification, transliteration, normalization and POS tagging of the dataset. [3] addressed the problem of shallow parsing of Hindi-English code-mixed social media text and developed a system for Hindi-English code-mixed text that can identify the language of the words, normalize them to their standard forms, assign them their POS tag and segment into chunks. [19] addressed

the problem of language identification on Bengali-Hindi-English Facebook comments. They annotated a corpus and achieved an accuracy of 95.76% using statistical models with monolingual dictionaries. [12] developed a Question Classification system for Hindi-English code-mixed language using word level resources such as language identification, transliteration, and lexical translation. [1, 16] performed Sentiment Identification in code-mixed social media text.
[18] proposed an algorithm for separating ironic from non-ironic similes in English, detecting common terms used in this ironic comparison. [8] presented a corpus of Italian tweets which consisted of 25.450 tweets among which 12.5% tweets were ironic and 87.5% tweets were non-ironic. They evaluated their dataset using two systems. The first system relies on lexical and semantic features characterising each word of a Tweet. The second system exploits words occurrences (BOW approach) as features useful to train a Decision Tree. [2] proposed a model to detect irony in English Tweets, pointing out that skipgrams which capture word sequences that contain (or skip over) arbitrary gaps, are the most informative features. [5] presented a corpus generated from review pairs on Amazon that can be used to identify sarcasm and irony in a tweet. [9] collected and annotated a set of ironic examples from a common collective Italian blog.

3 Corpus Creation and Annotation

In this section we explain the scheme used for corpus creation and annotation.

3.1 Corpus Creation

We constructed the Hindi-English code-mixed corpus using the tweets posted online since 2010. Tweets were scrapped from Twitter using the Twitter Python API which uses the advanced search option of twitter. We have mined the tweets using #irony, keywords 'irony' and 'ironic' and various hashtags from politics, sports and entertainment. The last three topics majorly but not essentially represent non-ironic tweets. As it is evident from example **T3** in section 1, it is not compulsory that irony is detected in all the tweets consisting of irony keywords and hashtags. We retrieved 1,19,885 tweets from Twitter in json format, which consists of information such as timestamp, URL, text, user, re-tweets, replies, full name, id and likes. An extensive semi-automated processing was carried out to remove all the noisy tweets. Noisy tweets are the ones which comprise only of hashtags or urls. Also, tweets in which language other than Hindi or English is used were also considered as noisy and hence removed from the corpus. Furthermore, all those tweets which were written either in pure English or pure Hindi language were removed, and thus, keeping only the code-mixed tweets. As a result, a dataset of 3055 code-mixed tweets was created. Newly created corpus and code is available online at Github.[1]

[1] https://github.com/deepanshu1995/Irony-Detection-Hindi-English-Code-Mixed-

3.2 Annotation

Annotation of the corpus was carried out as follows:

Language at Word Level : For each word, a tag was assigned to its source language. Three kinds of tags namely, 'eng', 'hin' and 'other' were assigned to the words by bilingual speakers. 'eng' tag was assigned to words which are present in English vocabulary, such as "Amazing", "Death", etc. 'hin' tag was assigned to Hindi words such as "sapna" (Dream), "hakikat" (Reality). The tag 'other' was given to symbols, emoticons, punctuations, named entities, acronyms, and URLs.

Ironic or Non-Ironic: : An instance of annotation is illustrated in figure 1. Each tweet is enclosed within <tweet></tweet>tags. First line in every annotation consists of tweet id. Language tags are added before every token of the tweet, enclosed within <word></word>tags. Each tweet is annotated with one of the two tags (Ironic or Non-Ironic). Irony is detected in 782 tweets. Remaining 2273 code-mixed tweets do not contain irony. The annotated dataset (consisting of tweet id's and annotated tag) with the classification system will be made available online later.

```
<tweet>
<id>831486289048457216<\id>
<word lang="eng">What</word>
<word lang="eng">an</word>
<word lang="eng">irony</word>
<word lang="other">?</word>
<word lang="hin">Jab</word>
<word lang="eng">relationship</word>
<word lang="hin">nai</word>
<word lang="hin">kiya</word>
<word lang="hin">tab</word>
<word lang="hin">sab</word>
<word lang="hin">Kuch</word>
<word lang="hin">mila</word>
<word lang="hin">Jab</word>
<word lang="eng">relationship</word>
<word lang="hin">mein</word>
<word lang="hin">hain</word>
<word lang="hin">tho</word>
<word lang="hin">Ek</word>
<word lang="hin">pic</word>
<word lang="hin">bhi</word>
<word lang="hin">nai</word>
<word lang="hin">mili</word>
<word lang="other">#ParSh</word>
<word lang="eng">Tales</word>
</tweet>
<class>
Ironic
</class>
```

Fig. 1. Annotated Instance

3.3 Inter Annotator Agreement

Annotation of the dataset to detect presence of irony was carried out by two human annotators having linguistic background and proficiency in both Hindi and English. A sample annotation set consisting of 50 tweets (25 ironic and 25 non-ironic) selected randomly from all across the corpus was provided to both the annotators in order to have a reference baseline so as to differentiate between ironic and non ironic text. In order to validate the quality of annotation, we calculated the inter-annotator agreement (IAA) for irony annotation between the two annotation sets of 3055 code-mixed tweets using Cohen's Kappa coefficient. Kappa score is 0.832 which indicates that the quality of the annotation and presented schema is productive.

4 System Architecture

In this section, we present our machine learning model for detecting irony in the code-mixed dataset described in the previous sections.

4.1 Pre-processing

Pre-processing of the code mixed tweets is carried out as follows. All the links and URLs are replaced with "URL". Tweets often contain mentions which are directed towards certain users. We replaced all such mentions with "USER". All the hashtags in the dataset are removed. All the emoticons used in the tweets are first stored to be used as a feature and then replaced with "Emoticon". All the punctuation marks in a tweet are removed. However, before removing them we store the count of each punctuation mark since we use them as one of the features in classification.

4.2 Classification Features :

In our work, we have used the following feature vectors to train our supervised machine learning model.

1. **Character N-Grams :** Character N-Grams are language independent and have proven to be very efficient for classifying text. These are also useful in the situation when text suffers from misspelling errors [10, 17, 20]. Group of characters can help in capturing semantic meaning, especially in the code-mixed language where there is an informal use of words, which vary significantly from the standard Hindi and English words. We use character n-grams as one of the features, where n vary from 1 to 3.
2. **Word N-Grams :** Bag of words feature is vital to capture the content in the text. Thus we use word n-grams, where n vary from 1 to 3 as a feature to train our classification models.

3. **Laugh Words and Emoticons :** Instead of using many exclamation marks internet users may use the sequence 'lmao' (i.e. laughing my ass of) or 'lol' (i.e. laughing out loud) or type hahaha. So we use a feature called laugh words which is the sum of all the internet laughs, such as 'haha', 'lol', 'lmao', 'rofl', 'lel', 'hehehe'. We also use emoticons as a feature for irony detection since they often represent textual portrayals of a writer's emotion in the form of symbols. We took a list of Western Emoticons from Wikipedia.[2]
4. **Punctuations :** Users often use exclamation marks when they want to express strong feelings. We count the occurrence of each punctuation mark in a sentence and use them as a feature.
5. **Intensifiers :** Users often tend to use intensifiers for laying emphasis on their feeling. A list of intensifiers was taken from Wikipedia. We count the number of intensifiers in a tweet and use the count as a feature.
6. **Negation words :** A list of negation words was taken from Christopher Pott's sentiment tutorial.[3] We count the number of negations in a tweet and use the count as a feature.
7. **Structure :** Ironical tweets in our dataset are often longer than other tweets. To capture this structure we use a group of features. (i) Number of characters present in the tweet. (ii) Number of words in the tweet. (iii) Average word length in the tweet.

Table 1. F1 Score for each feature using SVM classifier.

Features	F1 Score
All Features	0.77
Structural Features	0.64
Char N-Grams	0.77
Word N-Grams	0.70
Laugh Words + Emoticons	0.63
Punctuation Marks	0.63
Intensifiers	0.63
Negation Words	0.63

5 Experiments and Results

We performed experiments with two different classifiers namely Support Vector Machines with radial basis function kernel and Random Forest Classifier. Since the size of feature vectors formed are very large, we applied chi-square feature

[2] https://en.wikipedia.org/wiki/List_of_emoticons
[3] http://sentiment.christopherpotts.net/lingstruc.html

Table 2. F1 Score for each feature using Random Forest classifier.

Features	F1 Score
All Features	0.72
Structural Features	0.65
Char N-Grams	0.72
Word N-Grams	0.72
Laugh Words + Emoticons	0.63
Punctuation Marks	0.67
Intensifiers	0.63
Negation Words	0.63

selection algorithm which reduces the size of our feature vector to 1400[4]. For training our system classifier, we have used Scikit-learn [7]. In all the experiments, we carried out 10-fold cross validation. Table 1 and Table 2 describe the F1 score of each feature along with the F1 score when all features are used, in the case of Support vector machine and Random forest classifier respectively. Support vector machine performs better than Random forest classifier and gives a highest F1 score of 0.77 when all features are used. Character N-Grams proved to be most efficient in SVM, while word n-grams and character n-grams both resulted in best F1 score in the case of Random Forest Classifier.

6 Conclusion and Future Work

In this paper, we present an annotated corpus of Hindi-English code-mixed text, consisting of tweet ids and the corresponding annotations, which will be made freely available online later. We also present a supervised system used for detecting irony in the code-mixed text. The corpus consists of 3055 code-mixed tweets annotated as ironic or non-ironic. The features used in our classification system are character n-grams, word n-grams, emoticons, laugh words, punctuations, intensifiers and structural features. Best F1 score of 0.77 is achieved when all the features are incorporated in the feature vector using SVM as the classification system.

As a part of future work, the corpus can be annotated with part-of-speech tags at word level which could yield better results. Moreover, the annotations and experiments described in this paper can also be carried out for code-mixed texts containing more than two languages from multilingual societies, in future.

References

1. Aditya Joshi, Ameya Prabhu, Manish Shrivastava, and Vasudeva Varma: Towards Sub-Word Level Compositions for Sentiment Analysis of Hindi-English Code Mixed

[4] The size of feature vector was decided after empirical fine tuning

Text. In Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pp. 2482-2491. 2016.
2. Antonio Reyes, Paolo Rosso, and Tony Veale: A multidimensional approach for detecting irony in twitter. Language resources and evaluation 47, no. 1 (2013): 239-268.
3. Arnav Sharma, Sakshi Gupta, Raveesh Motlani, Piyush Bansal, Manish Srivastava, Radhika Mamidi, and Dipti M. Sharma: Shallow parsing pipeline for hindi-english code-mixed social media text. arXiv preprint arXiv:1604.03136 (2016).
4. Carol Myers-Scotton: Dueling Languages: Grammatical Structure in Code-Switching. Claredon. (1993).
5. Elena Filatova: Irony and Sarcasm: Corpus Generation and Analysis Using Crowdsourcing. In LREC, pp. 392-398. 2012.
6. Erik Forslid and Niklas Wikén. Automatic irony-and sarcasm detection in Social media. (2015).
7. Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel et al: Scikit-learn: Machine learning in Python. Journal of machine learning research 12, no. Oct (2011): 2825-2830.
8. Francesco Barbieri, Francesco Ronzano, and Horacio Saggion. "Italian irony detection in twitter: a first approach." In The First Italian Conference on Computational Linguistics CLiC-it, p. 28. 2014.
9. Gianti Andrea, Bosco Cristina, Bolioli Andrea, and Luigi Di Caro. "Annotating irony in a novel italian corpus for sentiment analysis." In 4th International Workshop on Corpora for Research on EMOTION SENTIMENT SOCIAL SIGNALS ES 2012, pp. 1-7. ELRA, 2012.
10. Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins: Text classification using string kernels. Journal of Machine Learning Research 2, no. Feb (2002): 419-444.
11. Kalika Bali, Jatin Sharma, Monojit Choudhury, and Yogarshi Vyas: "I am borrowing ya mixing?" An Analysis of English-Hindi Code Mixing in Facebook. In Proceedings of the First Workshop on Computational Approaches to Code Switching, pp. 116-126. 2014.
12. Khyathi Chandu Raghavi, Manoj Kumar Chinnakotla, and Manish Shrivastava: Answer ka type kya he?: Learning to classify questions in code-mixed language. In Proceedings of the 24th International Conference on World Wide Web, pp. 853-858. ACM, 2015.
13. Luisa Duran: Toward a better understanding of code switching and interlanguage in bilinguality: Implications for bilingual instruction. The Journal of Educational Issues of Language Minority Students 14, no. 2 (1994): 69-88.
14. Marjolein Gysels: French in urban Lubumbashi Swahili: Codeswitching, borrowing, or both?. Journal of Multilingual & Multicultural Development 13, no. 1-2 (1992): 41-55.
15. Pieter Muysken: Bilingual speech: A typology of code-mixing. Vol. 11. Cambridge University Press, 2000.
16. Souvick Ghosh, Satanu Ghosh, and Dipankar Das: Sentiment Identification in Code-Mixed Social Media Text. arXiv preprint arXiv:1707.01184 (2017).
17. Stephen Huffman. Acquaintance: Language-independent document categorization by n-grams. DEPARTMENT OF DEFENSE FORT GEORGE G MEADE MD, 1995.
18. Tony Vealy and Yanfen Hao: Detecting Ironic Intent in Creative Comparisons. In ECAI, vol. 215, pp. 765-770. 2010.

19. Utsab Barman, Amitava Das, Joachim Wagner, and Jennifer Foster: Code mixing: A challenge for language identification in the language of social media. In Proceedings of the first workshop on computational approaches to code switching, pp. 13-23. 2014.
20. William B. Cavnar, and John M. Trenkle: N-gram-based text categorization. Ann arbor mi 48113, no. 2 (1994): 161-175.
21. Yogarshi Vyas, Spandana Gella, Jatin Sharma, Kalika Bali, and Monojit Choudhury: Pos tagging of english-hindi code-mixed social media content. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 974-979. 2014.