ISSN: 2229-6959 (ONLINE) ICTACT JOURNAL ON SOFT COMPUTING, JANUARY 2017, VOLUME: 07, ISSUE: 02

DOI: 10.21917/ijsc.2017.0197

EMOTION AND SARCASM IDENTIFICATION OF POSTS FROM FACEBOOK DATA USING A HYBRID APPROACH

Raghavan V M[1], Mohana Kumar P[2], Sundara Raman R[3] and Rajeswari Sridhar[4]

Department of Computer Science and Engineering, College of Engineering Guindy, Anna University, Chennai, India

E-mail: [1]raghavan.eshwar94@gmail.com, [2]mohan1005k@gmail.com, [3]sundar24295@gmail.com, [4]rajisridhar@gmail.com

Abstract

Facebook has become the most important source of news and people's feedback and opinion about almost every daily topic. Facebook represents one of the largest and most dynamic datasets of user generated content. Facebook posts can express opinions on different topics. With this massive amount of information in Facebook, there has to be an automatic tool that can categorize these information based on emotions. The proposed system is to develop a prototype that help to come to an inference about the emotions of the posts namely anger, surprise, happy, fear, sorrow, trust, anticipation and disgust with three sentic levels in each. This helps in better understanding of the posts when compared to the approaches which senses the polarity of the posts and gives just their sentiments i.e., positive, negative or neutral. The posts handling these emotions might be sarcastic too. When detecting sarcasm in social media posts, the various features that are especially inherent to Facebook must be considered with importance.

Keywords:

Emotion, Sarcasm, Bipartite, Fuzzy, Conflicting Emotion Model

1. INTRODUCTION

Emotion detection and sarcasm detection from social net-working sites has been a great field of study. With the growth of e-services such as e-commerce, e-tourism and e-business, the companies are very keen on exploiting emotion and sarcasm analysis for their marketing strategies in order to evaluate the public attitudes towards their brand. Thus efficient emotion and sarcasm modeling system can be a good solution to the above problem. This work aims at developing a system that groups posts based on emotions and find sarcastic posts under each emotion, if present. This section is organized as follows: Section 2 discusses some existing work on social networking sites analysis, section 3 discusses the overall system architecture, section 4 discusses the modules in detail with algorithms, section 5 discusses the experimental results obtained and their evaluation and section 6 discusses the conclusion obtained from our work and future works.

2. RELATED WORK

Sentiment analysis is considered as a classification problem. In one of the works, the authors have claimed that classifier ensembles formed by diversified components are promising for tweet sentiment analysis [1]. The bag-of-words and feature based strategies are compared for the representation of tweets. In another work for sentiment analysis, all evaluation tests were performed with two classifiers, Maximum Entropy (Max-Ent) and Support Vector Machines (SVM) [2]. Emotion identification in text has become an interesting area of research work.

The identification of emotion on social media has gained lot of attention in recent years. This is required to find the opinions of people. Most of the works have been done by using training model. Researchers have automated the analysis of emotions in the text [3]. It is done by annotating the large dataset with six basic emotions namely anger, fear, disgust, joy, surprise and sadness. In another work, both speech and text is analyzed for detecting emotions along with intensities [4]. This approach uses the emotion modification words such as very and not which are considered to alter the emotion intensity. SenticNet [5] is used for opinion mining which is built using common sense reasoning techniques along with emotion categorization model. SenticNet discusses about concept level Semantic analysis [5]. Researchers have used combination of SentiWordNet and WordNet (WordNet Affect) to find the emotion in web based Content [6]. WordNet is used to find the similarity of text with emotion synsets. Words have different meaning depending on the context they appear which is not handled. This approach also analyses for patterns in the text which could be potential information for identifying variations in emotion intensity. An automatic rule- based system to extract the emotional cause even in Chinese micro blog posts is built and the classifier is trained to detect emotions present in Chinese micro blog posts based on the extracted emotion cause events [7]. Sarcasm detection has been done for online product reviews and tweets. For online product reviews [8], punctuation and pattern based features have been used. A semi supervised algorithm is applied on this, which in turn uses the results of k-nearest neighbor's algorithm. When compared to the online reviews, tweets are context less, which allows to detect sarcasm more efficiently just from individual sentences. Thus for tweets [9], the same algorithm was used, and it has shown better result due to the reason mentioned above. Another work is primarily focused on building a training model based on several features pertaining to tweets such as Mechanical Turk cues, Statistical cues, linguistic information, semantic information, length information [9]. But this approach had a lot of drawbacks. It is very costly to manually identify the cues related to each emotion. Also this approach needed a large training set to fetch the Statistical cues. The generalized rules for sarcasm based on the length information can't be applied to a diverse dataset such as Facebook. Thus, applying the same algorithm for Facebook posts will not return desired results since posts are long when compared with tweets and Facebook feature sets also vary. Since, sarcasm is user perspective analyzing the comments made to a Facebook post helps in identifying the sarcasm effectively. Most of the existing work for emotion detection and sarcasm detection is based on the Twitter dataset and its feature set [1], [6], [9], [10] and other regional micro blog datasets [2], [7].

Tweets are 140 character contents whereas Facebook posts can be of 60,000 characters. Thus, classifier used for building the emotion detection system should be tuned to the Facebook's feature set. Most commonly used classifiers such as SVM, Naive Bayes are based on a training corpus. Usage of classifiers based on training corpus is not efficient and accurate for text of large size. Moreover, Hashtag information is not taken into consideration in emotion detection [6] or sarcasm detection [10], although the words forming hashtags are useful keywords. In our work, we tried to accommodate the hashtag that the users use to express their emotions as a parameter in addition to the text based features for identifying emotions. The identified emotions along with other Facebook pertaining features like emoticons, likes, mood status, comments etc. have been used to identify the presence of sarcastic posts in Facebook and are discussed in the next section.

3. SYSTEM ARCHITECTURE

The block diagram of the entire system is shown in Fig.1. Facebook posts are retrieved using graph API. Posts which contain non-English words are filtered out. The posts are tagged using Stanford POS tagger and features pertaining to the post such as hashtags, special phrases, PS tags, emoticons, comments data are extracted from the posts.

The hashtag processor identifies the correct possible split up by generating an n-ary tree, which consists of all possible split up combinations.

The emotion identification module identifies the major emotion of the post by finding the similarity measure for each word in the post with that of the emotions followed by modification of similarity values based on pattern identification and polarity information. The emotion identification module uses the lexical databases WordNet and the SentiWordNet to find the right similarity scores for the words with respect to each emotion. The fuzzy union of all the similarity measure for the keywords is taken to obtain the measure for the entire sentence, the emotion with the maximum score is taken as the emotion of the sentence. The emotion score of the sentence is normalized based on the number of words in the sentence. The same methodology is applied to each of the sentences to find the emotion of the whole paragraph and then to the entire post text.

The emotions found are fed to sarcasm detection module. This module has three sub modules namely break points based sarcasm detection, special features based sarcasm detection and comments based sarcasm detection. All these sub modules use a conflicting emotion model to check against with the valid conflicts of emotions and thus find the score, described according to the model.

4. MODULE DESIGN

4.1 FECTHING OF SOCIAL FEEDS

The Graph API is the primary way to get data in and out of Facebook's social graph. It's a low-level HTTP-based API that you can use to query data, post new stories, upload photos and a variety of other tasks that an app might need to do. The application then uses the Graph API to fetch the payload from Facebook database. The payload is analyzed and the required fields are extracted.

The language in which the posts are written is analyzed and non-English posts are filtered out. The rest of the posts are stored in the database with corresponding data in the corresponding fields.

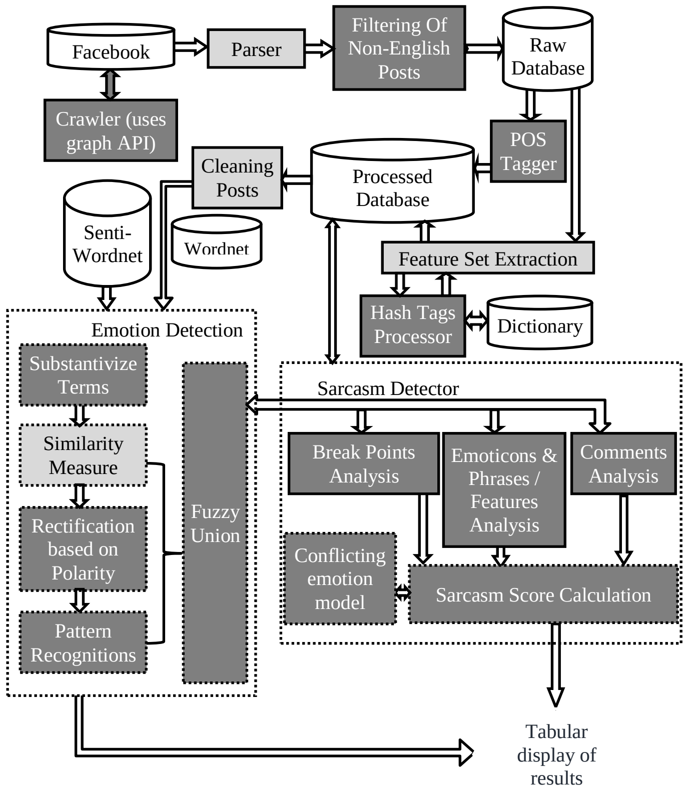

Fig.1. Block Diagram

4.2 POS TAGGER

The post text is analyzed and the Part-of-speech of every word in the text is found out using Stanford POS tagger. The context of the word in the sentence is also taken into account before finding the right Part-of-speech i.e., Context based evaluation is done for this module. Also special features of Facebook's posts such as hashtags must be properly identified and appropriately tagged so that they can handled in the required manner in the later stage.

Other POS taggers, like RiTA POS tagger was tried. However, the POS we were looking for like nouns, verbs, adjectives were found the same in both Stanford and RiTA POS tagger and hence we decided to go ahead with Stanford POS tagger as integration was easier.

For example, consider "He feels sad to eat food". By POS Tagging, we get: "PRP/He VBD/feels JJ/sad TO/to VB/eat NN/food/".

4.3 CLEANING POSTS

The posts from Facebook consists of a lot of noise such as http links, elongated words etc., The words which doesn't contribute in emotion or sarcasm detection, such as prepositions, pronouns, conjunctions are removed. From each keyword the Unicode characters are removed and stemming is performed.

4.4 FEATURE SET EXTRACTOR

Facebook posts have many features which makes it unique. The features include hashtags, emoticons, and special phrases like LOL, ROFL, RIP etc., POS tags, mood status and comments. These features are extracted and are used at various points of the system model.

Algorithm 1:
Algorithm for Hashtag Processing

```
1: str←GETINPUTSTRING
2: all_combination←FINDALLCOMBINATIONS
3: n←LENGTH(allcombinations)
4: scores ←null
5: for i 1 to n do
6:    measure = FINDRELATED (allcombination[i], post)
7:    scores = scores+(measure,i)
8: endfor
9: SORT(Scores)
10: returnALLCOMBINATIONS(scores[0],second)
```

4.5 HASHTAG PROCESSOR

In this work, we have contributed to process the hash tags and later use it for identifying emotion. Hashtags can give us some useful information. Usually Hashtag is given as a single word comprising of two or more meaningful words. The word(s) forming the hashtag also acts as important keywords for emotion and sarcasm modeling. E.g. #iphoneisgood to be replaced with "iphone is good'.

The algorithm for Hashtag processing is given in Algorithm 1. The single Hashtag information is given as input to the hash-tag processor, which creates an n-ary tree. We construct an n-ary tree with all possible words in the offline phase. During the online phase, in the n-ary tree, a root to leaf path contains the possible dictionary based split-up of words for the Hashtag information. After creating the n-ary tree, all valid root-to-leaf paths are traversed and valid split up words are gathered. Out of all split-up combinations obtained, the one with highest cosine similarity with that of the post is selected. Cosine similarity is used for text mining and since we had considered facebook data set with predominant text information, we used this similarity measure as against any other similarity metric. The keywords from the selected split-up combination are also utilized in emotion and sarcasm modeling. The hashtag sentence is converted into meaningful and relevant.

Ex: iphoneisgood: "iphone is good" and not as "iphone is good". somethingsunclear: "somethings unclear" and not as "somethingsunclear".

4.6 EMOTION IDENTIFIER

4.6.1 Finding the Substantivized Terms:

The emotion identifier module starts with finding a synset in the WordNet for each keyword. WordNet is a lexical database for English language, where each word is associated with synsets (a synset is set of synonyms). The selection of the appropriate synset is tricky, because the same word can have different meanings when used in different contexts. For example the noun "bank" can have different meanings, i.e.it can be a river bank or a monetary bank. So, in order to find a synset which correctly defines the sense in which the word is used in a particular sentence, we use the PYWSD (Python word sense disambiguation) package from NLTK [13].

In WordNet, the similarity can be calculated only between two noun synsets. So, words belonging to syntactical categories such as verbs, adverbs and adjectives are substantivized, i.e., a substantive noun synset is found, which is used in computing the similarity values. As given in Algorithm 2, for finding the substantive noun synset for a word, we find all possible synsets for the given word, for which there is match with the actual POS with the synsets POS. The most related noun of the derivationally related forms of the lemmas of these synsets is found, which is used as the substantivized noun synset for the word.

4.6.2 Similarity Measure:

For each of the 24 emotions, a synset from WordNet is associated, which is used for computing the similarity values. The similarity between the synset for each word and the synset for each emotion is computed with Wuand Palmer similarity algorithm.

In WuandPalmer methodology, the similarity score between two synsets $s1$ and $s2$ are found using the Eq.(1). For finding Wu-Palmer similarity between two nouns we are using the python nltk function[14].

$$Score = \frac{2 * depth(lcs)}{depth(s1) + depth(s2)}, 0 \le Score \le 1 \qquad (1)$$

4.6.3 Polarity based Correction:

This module uses the Senti WordNet resource. Senti WordNet is a lexical resource for sentiment analysis where in it provides numerical sentiment (positive, negative and objective) scores for a synset. Based on the numerical values, it is possible to classify whether a word belongs to positive or negative or objective sentiment.

The similarity values for a word can be same or similar for some opposite emotions. For example, the word sorrow has minute differences in similarity values for the emotions joy and sadness i.e, similarity between sorrow and joy is 0.75 and similarity between sorrow and sadness is 0.72. This is because of using Wu-Palmer similarity, which computes similarity score based on the path length, where synsets which are at same distance from their ancestors will have same or similar similarity scores.

For correcting this, we have classified the 24 emotions into two categories based on their sentiment. Thus, emotions such as joy, surprise, trust & anticipation and their sentic levels are taken as emotions with positive polarity. Similarly, emotions such as sadness, fear, anger, disgust and their sentic levels are taken as emotions with negative polarity. Thus, if the polarity of the word is different from the polarity of the emotion, then the similarity value for that emotion is complemented with the similarity value of the emotion with opposite polarity. After polarity based correction, the similarity between sorrow and joy is 0.28 and similarity between sorrow and sadness is 0.72.This method of correcting similarity scores based on polarity is incorporated from the research work in [6] and extended for our feature set.

```
Algorithm 2:
Algorithm for Substantivizing terms
1: synsets←DICTIONARY(word,actual_pos)
2: lemmas←null
3: derivationally_related_forms←null
4: for each synset in 1synsets do
5:     for lemma in each synset do
6:         if synset_pos = actual_pos then
7:             lemmas = lemmas+lemma
8:         endif
9:     endfor
10: endfor
11: for lemma in each lemmas do
12:     derivationally_related_forms+=
        (lemma,derivationally_related_forms)
13: end for
14: for each drf in derivation ally related forms do
15:     for each lemma in drf[1]
16:         if lemma_pos=NOUN then
17:             Noun _lemmas=Noun _lemmas
+lemma
18:         end if
19:     end for
20: endfor
21: returnmost relatednoun
```

4.6.4 *Pattern Analysis:*

In this phase, we identify the possible patterns in the text for modifying the similarity values. With the help of POS tagged information, we find patterns like RB-RB-JJ, RB-JJ etc. We maintain three types of word lists namely negators, positive intensifiers and negative intensifiers. The words such as ‘not’, ‘no’, ‘none’, ‘never’ etc., are negators which negate the emotion. The words such as ‘especially’, ‘exceptionally’, ‘excessively’, ‘extremely’, ‘extraordinarily’, etc.., are positive intensifiers which increase the emotion level of the word. The words such as ‘barely’, ‘moderately’, ‘slightly’, ‘less’, etc., are positive intensifiers which decrease the emotion level of the word. Thus, after finding the patterns we analyze the patterns to find out if it contains any negators, positive intensifiers, negative intensifiers or a combination of negators and intensifiers and change the similarity score based on the following formulae,

$$\mu(\text{negator}(word)) = 1 - Score(word) \quad (2)$$

The above Eq.(2) shows the modification of similarity scores on the presence of negators.

$$\mu(\text{pos\_intensifier}(word)) = \sqrt{Score(word)}\ \forall Score(word) \geq 0.5 \quad (3)$$

$$\mu(\text{pos\_intensifier}(word)) = \text{Score(word)}^2\ \forall\ Score(word) < 0.5 \quad (4)$$

The Eq.(3) and Eq.(4) shows the modification in the presence of positive intensifiers.

$$\mu(\text{neg\_intensifier}(word)) = Socre(word)^2\ \forall Score(word) \geq 0.5 \quad (5)$$

$$\mu(\text{neg\_intensifier}(word)) = \sqrt{Score(word)}\ \forall Score(word) \geq 0.5 \quad (6)$$

The Eq.(5) and Eq.(6) shows the modification on the presence of negative intensifiers.

$$\mu(\text{combinator}(word)) = \sqrt{Score(word) * \mu(intensifier(word))} \quad (7)$$

```
Algorithm 3
Algorithm for Emotion Identification
1:  patterns ← (RB − RB − JJ, RB − JJ)
2:  N ← numof paragraphs
3:  for  i in 1 to N do
4:      S ← NUMOFSENTENCES(paragraphs[i])
5:      for j in 1 to S do
6:          keyWords = ExtractKeywords(sentences[j])
7:          keyWords = NOUNIFY(keyWords)
8:          similarityScores = null
9:          polarity = null
10:         k ← 0
11:         for ite in keywords do
12:             polarity[k] = GETPOLARITY(ite)
13:             similarityTableWords[k]   =
                    SIMMEASURE(ite,
                polarity[k])
14:             k = k + 1
15:         end for
16:         IDENTIFY_PATTERNS
17:         MODIFY_TABLE
18:         similarityTableSentence [] =FUZZY
        UNION(SimTableWords, SimTableSentence)
19:         similarityTableSentence [] =
        ALPHACUT(SimTableSentence)
20:     end for
21:     similarityTableFinal [] =FUZZY
        UNION(SimTableSentence, SimTableFinal)
22: end for
23: similarityTableFinal []
      =ALPHACUT(SimTableFinal)
```

The Eq.(7) shows the modification on the presence of combinators, i.e. negator followed by either positive intensifier or negative intensifier. The Eq.(2) and Eq.(7) are incorporated from the research work done in [6].

4.6.5 *Fuzzy Union of Similarity Measures:*

The use of Fuzzy logic in the modeling of the emotions is a remarkably simple way to process vague, ambiguous or imprecise information. The fuzzy union of all the similarity measure for the keywords is taken to obtain the measure for the entire sentence; the emotion with the maximum score is taken as the emotion of the sentence. The emotion score of the sentence is normalized based on the number of words in the sentence. The same methodology is applied to each of the sentences to find the emotion of the whole paragraph and then to the entire post text. After computing the similarity table for a sentence the values in table which are less than 0.2 are removed which is mentioned as

ALPHACUT and this is also applied for the final similarity table and the details are given in the Algorithm 3. ALPHACUT is incorporated from the research work in [6].

4.7 SARCASM DETECTOR

After finding the eight major emotions, the sarcasm in the posts under each of these emotions are also detected using the following methodologies which also makes use of the emotion model. In this work, we have designed 3 methodologies for identifying sarcasm and is given in Algorithms 4, 5, 6.

4.7.1 Break Points Based Detection:

The various break points in a Facebook post include

- Conjunctions in the post.
- End of sentences.
- End of paragraphs.
- PS tags

These break points are found out and the entire Facebook post is separated into two parts with these breakpoints. The emotions of both the parts are analyzed for conflict with each other as in Algorithm 4.

Algorithm 4

Algorithm for Sarcasm Detection-Phase1

```
1: ite←Iteratorofthepost
2: W←Wordsinthepost
3: Break[]=(Conjunction, SentenceEnd,
            ParagraphEnd, PS)
4: for ite inPostdo
5:     flag←0
6:     for i in 1 to 4 do
7:         if itematchesBreak[i] then
8:             flag←1
9:             break
10:        endif
11:    endfor
12:    if flag==1 then
13:      if((FINDEMOTION(Post_begin,Post_begin
           + ite) != (FINDEMOTION(Post_begin
           + ite,Post_end))) then
14:            ASSIGNWEIGHT
15:      endif
16:    endif
17: endfor
```

4.7.2 Special Features Based Detection:

The special features include

- Emoticons
- Hash tags
- Special Phrases

The major emotion exhibited by these special features is found out based on the algorithm discussed in Algorithm 5. Also, the text part of the post is extracted separately and its emotion is found out. A valid conflict between these two emotions is checked.

4.7.3 Comments Based Detection:

Posts may or may not contain comments. But if they have one or more comments, the major emotions exhibited by these comments are found out. If they differ from the emotion exhibited by the post, we claim that the post is said to be sarcastic. Algorithm 6 is utilized for this sub-module.

4.7.4 Level of Sarcasm-Conflicting Emotion Model:

The Plutchik's wheel of emotion is shown in Fig.2 [12]. The eight basic emotions constitute the region between the first and second circle. There is an intensified and diluted emotion for each of the eight emotions totaling to 24 emotions.

Algorithm 5

Algorithm for Sarcasm Detection-Phase2

```
1: postEmotion=FINDEMOTION(Post)
2: EmoticonsEmotion=null
3: PhrasesEmotion=null
4: HashTagsEmotion=null
5: H←Hashtags in the post
6: E←Emoticons in the post
7: P←Special phrases in the post
8: h←Number of hashtags
9: e←Number of emoticons
10: p←Number of special phrases
11: Emotions=[]
12: for i in 1 to e do
13:    Emotions=Emotions+FINDEMOTION (E[i])
14: endfor
15: EmoticonsEmotion=FINDMAJOREMOTION
                     (Emoticons)
16: Emotions=[]
17: for i in 1 to p do
18:   Emotions=Emotions+FINDEMOTION (P[i])
19: endfor
20: PhrasesEmotion=FINDMAJOREMOTION
                   (Phrases)
21: Emotions=[]
22: for i in 1 to h do
23:    Emotions=Emotions+FINDEMOTION (H[i])
24: endfor
25: HashTagsEmotion =FINDMAJOREMOTION
                     (Hashtags)
26: if((EmoticonsEmotion!= WordsEmotion) OR
      (PhrasesEmotion!= WordsEmotion) OR
      (HashTagsEmotion != WordsEmotion)) then
27:     ASSIGN_WEIGHT
28: endif
```

The sarcasm score calculation is based on the conflicting emotion model shown in Fig.3. The model is built using the 8 emotions with 3 sentic levels. The model is a Bipartite Graph with each emotion and its levels collectively as a node. The emotions Joy, Anticipation, Anger and Trust comes under one set, say 'A', whereas the emotions Fear, Surprise, Disgust and Sadness comes under another set, say ‘B’. There are no conflicts amongst the emotions of each set which makes this graph a bipartite one. The edges represent conflict between respective emotions (belonging to two sets, one from each) between two constituents of the post. The weight denotes the capacity of the edge.

The level of sarcasm is calculated as follows. A conflict can occur between any sentic level of one emotion with any sentic level of another emotion.The node weight of the sentic level ranges from 3 to 1, 3 being assigned to the most intense sentic level of an emotion. The weight of an edge(conflict) is calculated by adding both the nodes’ weight. A particular emotion in any set, can have a conflict with at most two emotions of the other set, and with at most 3 sentic levels of each of those conflicting emotions.

Algorithm 6
Algorithm for Sarcasm Detection-Phase 3

```
1: C←FETCHCOMMENTS(Post)
2: Emotions=[]
3: for c in 1 to size(C) do
4:     Emotions=Emotions+FINDEMOTION(C[i])
5: endfor
6: CommentsEmotion=FINDMAJOREMOTION
                        (Emotions)
7: if CommentsEmotion!=WordsEmotion then
8:     ASSIGNWEIGHT
9: endif
10: return CALCSCORE
```

If the conflicting emotion is exactly opposite to it in the Plutchik wheel of emotions, it contributes to the sarcastic score twice the amount (denoted by thick lines in Fig.3) than the contribution done by the conflicting emotions which are not exactly opposite (denoted by thin lines in Fig.3). Also, there is no necessity that all the possible conflicting pair of emotions must be occurring in the post text. Only when some particular emotion occurs, its conflicting counterparts are looked upon for occurrence, otherwise the conflicting pairs are neglected. Based on the number of conflicting pairs that occurs, the sarcasm score for the post is normalized and the score ranges from zero to hundred.

For example, let us suppose that, there are only two conflicts arising combining all the three methodologies; one between ecstasy and grief (edge weight is 6 and individual contribution is 100% as the maximum possible edge weight of an edge is 6), and the other between ecstasy and fear (edge weight is 5 and individual contribution is 83.33%). The conflict between ecstasy and grief contributes to double the amount as the contribution made by the conflict between ecstasy and fear, as the former pair lies exactly opposite in the Plutchik wheel of emotions. Thus the final sarcastic score would be 94.44%.

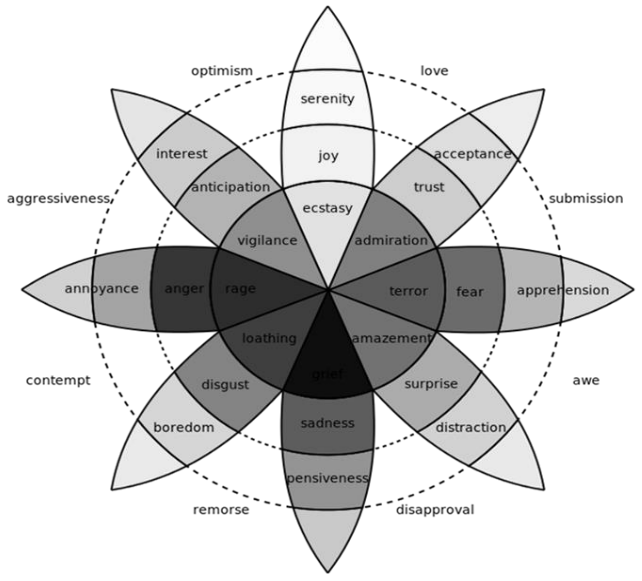

Fig.2. Plutchik Wheel of Emotions

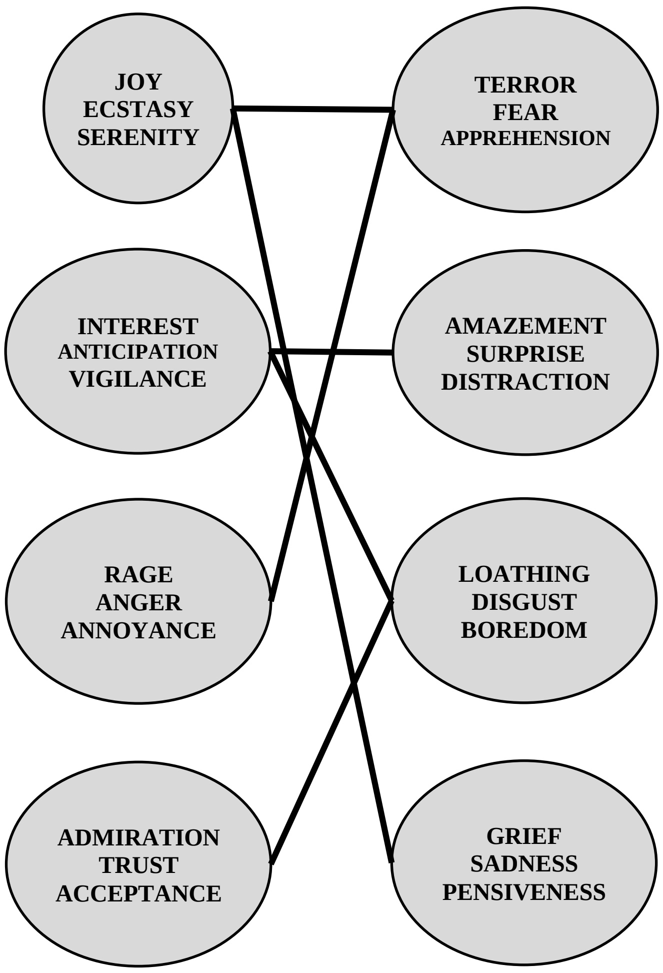

Fig.3. Conflicting Emotion Model

5. RESULTS AND EVALUATION

5.1 DATA SET DESCRIPTION

Posts are extracted from various public pages, groups and profiles using the GRAPH API. The extracted posts are in JSON format. The JSON format of a post comprises of several attributes such as 'message' of the post, 'user id' of the post which uniquely identifies the user who posted it, 'post id' which uniquely identifies the post, 'created time' , 'updated time', 'comments' , 'type' etc. These posts are then processed to filter out non English posts. The remaining posts are input to the system

5.2 EVALUATION

"Precision" is defined as the ratio between true positives and sum of true positives and false positives; "Recall" is defined as the ratio between true positives and sum of true positives and false negatives. Here,

True positives - correctly identified/classified post

False positives - incorrectly identified/classified post

True negatives - correctly ignored post

False negatives - incorrectly ignored post

Recall is calculated as follows:

$$Recall = \frac{True\ Positives}{True\ Positives + False\ negatives} \tag{8}$$

i.e, Recall = No. of correctly classified posts/No. of posts manually classified

Precision is calculated as follows:

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \tag{9}$$

i.e, precision = No. of correctly identified posts/Total no. of posts identified.

F-Measure is calculated as follows:

$$F\ Measure = \frac{2 * Precision * Recall}{Precision + Recall} \tag{10}$$

$$Accuracy = \frac{True\ Positives}{True\ Positives + False\ Positives} \tag{11}$$

Precision expresses how efficient and accurate the algorithm behaves in grouping the posts and recall expresses the ability to apply the algorithm on all possible posts. F-score is a single measure that combines precision and recall. Our aim is to have high precision, recall and hence an increased F-score. Recall could be compromised as the algorithm may not have learnt all possible scenarios and hence we are more particular about a higher precision in grouping and classifying the posts.

We calculated precision, recall and F-score for emotion identification and sarcasm identification as they are the basis for grouping the posts. The Table.1 and Fig.4 shows the precision, recall and F-score of Emotion identification. As discussed already we have grouped 24 emotions into 8 groups and have tabulated the results. As can be seen, the emotion under the "trust" category has a lower precision. The dataset considered is from social media and hence we will not be in a position to rely on the authenticity of all its content. However, other emotions had higher precision. The features considered as discussed in the algorithm contributed to this increased precision and recall. The recall of all the emotions is almost same and that justifies the efficiency of the algorithm's learning and retention.

Table.1. Precision, Recall and F-measure for Emotion Identification

Emotions	Precision	Recall	F measure
joy	0.82	0.71	0.76
sadness	0.81	0.71	0.76
fear	0.82	0.75	0.78
anger	0.85	0.73	0.78
trust	0.62	0.74	0.67
anticipation	0.72	0.70	0.71
surprise	0.88	0.80	0.84
disgust	0.72	0.77	0.74

In sarcasm identification, each post is given a sarcastic score that ranges from 0 to 1. Posts that have scores above 0.2 are considered as sarcastic and those posts are alone displayed as results. Remaining posts are discarded. The Table.2 and Fig.5 indicate the precision, recall and f-score for sarcasm identification.

From the above tables and charts, we infer that our proposed system produces results which have an average precision value of 82%. This implies that the system is efficient in identifying the right emotion for the post and sarcastic nature of the post. There is a drop in precision value for 'trust' which can be attributed to the fact that many posts which don't exhibit any major emotion falls into the 'trust' category.

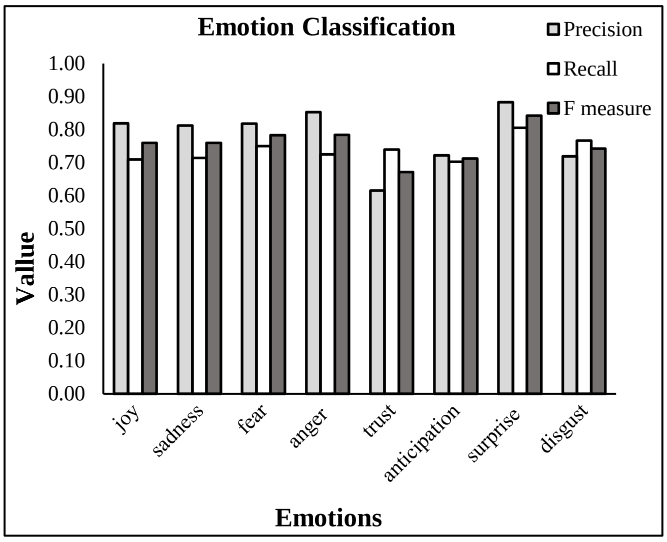

Fig.4. Precision, Recall and F-Measure for Emotion Identification

This is because most of the words have some similarity with ‘trust’ emotion synsets and the fuzzy union of similarity measures of those words makes the system to categorize those posts under ‘trust’. The dip in recall values is because of variations in the identification of similarity measures of words in the post. Since our system uses Wu Palmer’s Similarity algorithm, words which are having relatedness with ‘Joy’ will also have relatedness with ‘Surprise’. As we are finding the major emotion, these variations in similarity measures get bloated up when the post is taken as a whole and hence the reduced recall values.

Our system identifies the sarcastic score of the post using the conflicting emotion model. When a post doesn’t have break points or if it doesn’t contain any special phrases or if the emotion exhibited by the comments is not conflicting to that of the post’s emotion or if there are not any comments our system fails to identify the sarcastic score of the post. This results in the dip in the recall values.

Table.2. Precision, Recall and F-measure for Sarcasm Detection

Emotions	Precision	Recall	F measure
joy	0.74	0.63	0.68
sadness	0.71	0.60	0.65
anger	0.73	0.67	0.70
trust	0.69	0.67	0.68
surprise	0.74	0.65	0.70
disgust	0.61	0.79	0.69

Also in our dataset, there were no posts which were sarcastic and at the same time coming under the emotions fear and anticipation. It allows us to conclude that posts which exhibit the emotions fear and anticipation are mostly not sarcastic. So precision, recall values for those emotions were not shown in the sarcasm detection graph.

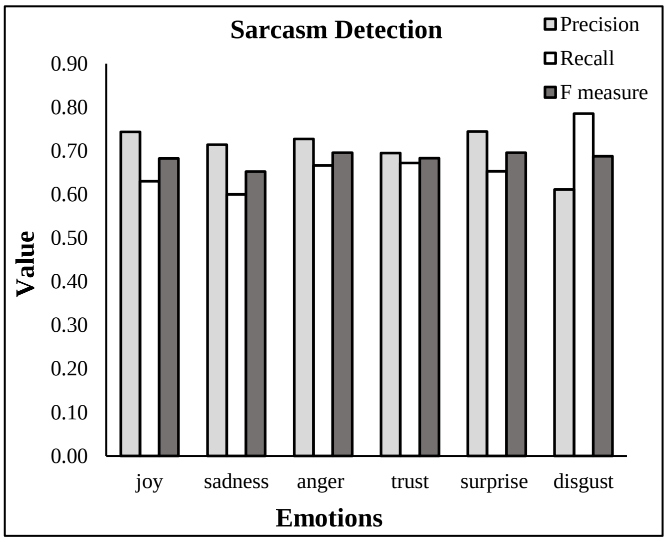

Fig.5. Precision, Recall and F-Measure for Sarcasm Detection

The Table.3 shows the evaluation metrics for Hashtag Processing module. Every word(s) which is a hash tag is correctly identified by the system as a Hashtag. Thus the true negatives and false positives are not considered in the evaluation. The other two metrics are utilized to find the accuracy (Eq.(11)) of correctly identifying the combination of words in a hash tag. Our system finds the proper combination with an accuracy rate of 87.14%. Drop in accuracy to 87% is due to the presence of improper non-dictionary words in hashtag contents and improper nouns, such as names of organizations which doesn’t occur in the post’s text content.

Table.3. Evaluation for Hashtag Processing

Number of hash tags (2000 Posts)	True Positives	False Positives	Accuracy
1073	935	138	0.87

6. CONCLUSION AND FUTURE WORKS

The statistical analysis obtained via feedback from end users shows that the developed system can identify the right emotions and detect sarcasm in posts that are grammatically correct and contain emotion exhibiting words. Also this type of model is applied for the first time to Facebook posts which is comparatively large and diverse in terms of feature sets. The system also considers hash tags which are an important feature set of Facebook posts. The sarcasm identification model is a novel approach based on emotion model.

The errors in the POS Tagger propagate down to all the modules of the system. Most of the errors in the system have been found to be due to incorrect POS. Also the sarcasm detector is too much dependent on the emotion identification module which poses risk at times. The performance evaluation of the system reveals the inability of the system to handle the variations in similarity score computation which in turn leads to reduced recall values. The system also fails to find sarcastic posts which don’t satisfy the conditions in the sarcasm identification model.

The efficiency of Stanford POS tagger is only around56% for sentences. Hence, it could be replaced by a better POS tagger to improve the efficiency. A technique to detect sarcasm where there is no conflict of emotions is to be identified. Future extensions to this work can also take into account the slang details while detecting sarcasm. Framing new rules by analyzing reactions (which has been newly added to Facebook instead of just the “Like” button) for posts to detect level of sarcasm effectively can also be done. Thus, there are a lot of improvements and extensions possible and this system offers a wide scope in the field of Natural Language Processing.

REFERENCES

[1] Nadia F.F. da Silva, Eduardo R. Hruschka and Estevam R. Hruschka Jr., “Tweet Sentiment Analysis with Classifier Ensembles”, *Decision Support Systems*, Vol. 66, pp. 170-179, 2014.

[2] Ivan Habernal, Tomas Ptacek and Josef Steinberger, “Supervised Sentiment Analysis in Czech Social Media”, *Information Processing and Management*, Vol. 50, No. 5,

pp. 693-707, 2014.

[3] C. Strapparava and R. Mihalcea, “Learning to Identify Emotions in Text”, *Proceedings of the 2008 ACM Symposium on Applied Computing*, pp. 1556-1560, 2008.

[4] Ze-Jing Chuang and Chung-Hsien Wu, “Multi-Modal Emotion Recognition from Speech and Text”, *Computational Linguistics and Chinese Language Processing*, Vol. 9, No. 2, pp. 45-62, 2004.

[5] E. Cambria, R. Speer, C. Havasi and A. Hussain, “SenticNet: A Publicly Available Semantic Resource for Opinion Mining”, *Commonsense Knowledge: AAAI Fall Symposium*, Vol. FS-10-02, pp. 14-18, 2010.

[6] Vincenzo Loia and Sabrina Senatore, “A Fuzzy-Oriented Sentic Analysis to Capture the Human Emotion in Web-based Content”, *Knowledge-Based Systems*, Vol. 58, pp. 75-85, 2014.

[7] Weiyuan Li and Hua Xu, “Text-based Emotion Classification using Emotion Cause Extraction”, *Expert Systems with Applications*, Vol. 41, No. 4, Part 2, pp. 1742-1749, 2014.

[8] O. Tsur, D. Davidov and A. Rappoport, “ICWSM-A Great Catchy Name: Semi-Supervised Recognition of Sarcastic Sentences in Online Product Reviews”, *Proceedings of the Fourth International Conference on Weblogs and Social Media*, pp. 162-169, 2010.

[9] R. Gonzlez-Ibez, S. Muresan and N. Wacholder, “Identifying Sarcasm in Twitter: A Closer Look”, *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers*, Vol. 2, pp. 581-586, 2011.

[10] Raquel Justo, Thomas Corcoran, Stephanie M. Lukin, Marilyn Walker and M. Ines Torres, “Extracting Relevant Knowledge for the Detection of Sarcasm and Nastiness in the Social Web”, *Knowledge-Based Systems*, Vol. 69, pp. 124-133, 2014.

[11] Benno Stein and Sven Meyer Zu Eissen, “Document Categorization with MajorClust”, *Proceedings of 12th Workshop on Information Technology and Systems*, 2002.

[12] Robert Plutchik, “The Nature of Emotions”, *American Scientist*, Vol. 89, No. 4, pp. 344-350, 2001.

[13] Liling Tan, “Pywsd: Python Implementations of Word Sense Disambiguation (WSD) Technologies [Software]”, https://github.com/alvations/pywsd, 2014.

[14] Steven Bird, Ewan Klein and Edward Loper, “*Natural Language Processing with Python*”, O’Reilly, 2009.