Construct-Based Sentiment Analysis Model

Smriti Singh, Jitendra Kumar Rout and Sanjay Kumar Jena

Abstract Expression of opinions is basic human nature. With the advent of various online platforms, this expression has largely taken digital form. More often than not, it is in the interest of enterprises and individuals to know the sentiment of these opinions, be them in the form of reviews, blogs or articles. Given the humungous amount of this data, it becomes essential to analyze it programmatically and with accuracy. The paper looks at various methods of doing this and also suggests one which takes into account the sentence constructs and the way the sentences are framed. One of the primary concerns is also to detect and handle negations and contradictions occurring in the sentences.

Keywords Opinion mining · Sentiment analysis · Natural language processing

1 Introduction

Starting with a movie review polarity dataset, containing 5331 positive and 5331 negative processed sentences shown in Table 1, which were manually tagged as positive or negative. The sentences are first POS-tagged using HMM and Viterbi method. A dictionary-based approach is used using SentiWordNet. SentiWordNet is a device that is generally utilized as a part of sentiment mining, and is focused

S. Singh (✉) · J.K. Rout · S.K. Jena
National Institute of Technology, Rourkela, Sundargarh 769008, Odisha, India
e-mail: smritisingh22@gmail.com

J.K. Rout
e-mail: jitu2rout@gmail.com

S.K. Jena
e-mail: skjena@nitrkl.ac.in

D.K. Lobiyal et al. (eds.), *Proceedings of the International Conference on Signal, Networks, Computing, and Systems*, Lecture Notes in Electrical Engineering 396, DOI 10.1007/978-81-322-3589-7_18

Table 1 The movie review dataset

Total number of tweets	10,662
Number of positive tweets	5331
Number of negative tweets	5331

around an English lexical lexicon called WordNet [1]. We extract individual sentiment scores and summate them to get overall polarity of a sentence. This method gives unsatisfactory results and thus we move on to machine learning-based approaches. We start with two baseline methods to find the sentiment of a test sentence, using the Naiye bayes method and the K-nearest neighbour method.

1.1 Naive Bayes Method

The Naive Bayes classifier, based on Bayes theorem is a simple probabilistic classifier with strong and naive independence assumptions, i.e. the occurrences of entities are independent of each other. It is commonly used in email spam detection and sorting, sentiment detection, categorization of electronic content. This classifier is very efficient even though it is outperformed by techniques, such as max entropy, support vector machines, etc, since it is less computationally intensive (both in terms of CPU and memory usage). It requires a small amount of training data also the training time is significantly smaller as opposed to alternative methods [4]. The method assigns a score of negativity and positivity to each word based on the frequency of its occurrence in a positive or a negative context, respectively. To score a document 'd' for a class 'c' having words 'w', conditional probability:

$$P(c|d) = \text{argmax}[P(c) * \prod_{i=1}^{n} P(w_i|c)] \tag{1}$$

We choose the label with the highest probability.

Sentiment Calculation The method assigns a positivity and a negativity score to each word in the sentence based on the frequency of its occurrence in a negative or positive context. The prior probability $P(c)$ of a class c: (c could be positive or negative)

$$P(c) = \text{no. of words in } 'c'/\text{total no. of words in the corpus} \tag{2}$$

In general, sentiment value associated with each word(w) = [pos_score(w)− neg_score(w)]. The sentiment of the whole sentence is calculated as the summation of sentiment values of its of its constituent word.

Table 2 A comparision of Naive Bayes and K-NN method on move

Training set data size (%)	Test set data size (%)	Naive Bayes (%)	K-NN algorithm (%)
90	10	79.00	68.07
80	20	77.90	67.35
70	30	76.60	67.14

1.2 K-NN Method

K-NN is a lazy algorithm that determines the sentiment of a sentence by checking the sentiment of *k*-sentences which are the closest neighbours of the given sentence (the distance metric being the number of matching words) and taking the sentiment which majority of these neighbours have. *K* was chosen to be five for our experiment. A comparison of the two methods is shown in Table 2.

Drawbacks in these two methods are that each word is considered independent of the other, while words affect each other.

The rest of the paper is organized as follows: Sect. 2 discusses the work previously done with relation to *n*-gram model. Section 3 describes the proposed construct-based model in details. Section 4 describes the detailed algorithm used as well the results obtained. Section 5 concludes the work done, highlighting the contributions and suggests directions for possible future work.

2 Related Work

The unigram model assumes that all good words in a sentence can occur independently of any other words in the sentence [7]. In basic sentences with sentiment heavy words, or where words qualify their sentiments individually, this method works fine. Example"I disliked the movie.", "The book is nice.", etc. However, the method fails to give good results if the works are related and change the sentiments of other words, as happens when we use a negative word in a sentence. Example: "I did not like the movie."

n-gram: A slightly improvised version of the *n*-gram model is to use a bigram, where we take words in groups of two [3, 7]. This improves the accuracy level slightly but doesn't give any impressive result. Tri-gram method completely fails giving bad accuracy for the dataset we use. A combination of bi-gram and uni-gram model using Nave Bayes method works better than other methods. Table 3 depicts some of the comaprative results.

But we conclude that using an *n*-gram model does not provide a solution to our problems such as negation words and sentiment neutralisers. Thus, we model a construct-based method discussed in the next segment to solve our issue.

Table 3 *n*-gram method

# of Training Sentences (Total = 5330)	70 % (3730)	80 % (4265)	90 % (4800)
Unigram	76.32	71.79	77.04
Bi-gram	77.53	73.49	78.42
Unigram + Bi-gram	78.91	73.07	79.85

3 The Construct-Based Model

As a human reader, it comes to us naturally to determine the overall opinion or sentiment of a sentence. Sentence opinion is a function of the opinions of the individual constituents of a sentence. Classifying sentiment on the basis of individual words can give misleading results because atomic sentiment carriers can be modified (weakened, strengthened, or reversed) based on lexical, discoursal or paralinguistic contextual operators [8].

Past attempts to deal with this phenomenon include writing heuristic rules to look out for negatives and other changing words [6], combining the scores of individual positive and negative word frequencies [12], and training a classifier on a set of contextual features [11].

Several rules govern the polarity of words or group of words in a sentence [5, 9, 10]. In Neviarouskaya et al. 2010 [5], six composition rules were defined, i.e. domination, neutralization, sentiment reversal, aggregation, propagation, and intensification. Our model is an extension of the same idea. We propose to model these rules in a construct-based model for polarity determination. We use the basic Naive Bayes classification method as the classifier.

The proposed essential rules that have been implemented in our model are as follows:

1. *Parts of Speech that provide sentiment information* Certain Parts of Speech are better indicators of opinions than the others. The sentiment is mainly determined by the order in which these certain parts of speech occur in a sentence. In this process, we skim over words that do not provide any useful information about the sentiment of a sentence. Thus, in our model, we determined those parts of speeches and calculated polarity for the ones that are indicative and contributors of sentiments in a sentence, such as, adjective, noun, verb, and adverb.
2. *Neutraliser* If a compound sentence has connecting words such as "but," "nevertheless," "still," we observe that the sentence part following the connector has a domination over the sentiment of the overall sentence. Hence, during calculations, we neutralize to sentence polarity till the point we observe the connecting word and do the calculations following it. Example: "The phone is costly but it has amazing features." This is applicable only when the words are used as conjunctions. For example, But is a three-lettered word is not a valid example for this construct.

3. *Intensifier* The rule of intensification is applied to strengthen or weaken a sentiment polarity. Examples of intensifiers are words such as "very" and "extremely." These are valence shifters which have the ability to increase or decrease the sentiment polarity of a word. We divide them into two categories: Incrementors and Decrementors, where Incrementors intensify and decrementors weaken the sentiment score of the corresponding word(s) they refer to. As an example, the sentence "I am extremely happy." will have the (positive score of happy) > (positive score of extremely happy). We have a list of pre-determined opinion incrementors and decrementors which have the ability to intensify or diminish the word sentiment polarity. Examples of decrementors would be reduced, decreased, etc.
4. *Negation* Once individual word sentiments are determined as per the methods described in the previous sections, the next step is to determine how to use these values to handle negations. As a human, in the case of occurrence of a negation, we can easily determine, what it is, that is being negated. Our model tries to simulate this behaviour to handle the analysis of sentiments in the sentence and any negations that may occur. The main problem with handling simple sentences with negation is to determine the scope of effect of the negative word. Traditional methods include reversing polarity till the end of sentence. Some other methods include, scope of negation being words between negation and first punctuation mark [7]. The works described in [2, 3] suggest scope of negation to be next five words.
   The approach aims at inverting sentiments in the vicinity of the negation words. In our proposed model, we determine this vicinity in terms of, what we call, the depth of negation. In a crude sense, depth of negation is the number of words around the negator whose polarity should be flipped. While calculating this depth, we noticed certain sentence constructs that occur frequently in negative sentences. Example negation sentence phrases and their corresponding POS tags:

"do not buy": VB−NEG−VB	"be not worth": VB−NEG−JJ
"would not recommend": MD−NEG−VB	"not the good": NEG−DT−JJ
"not very impressed": NEG−RB−JJ	"not bad": NEG−JJ
"not a good choice": NEG−DT−JJ−NN	

5. *Delimiter* A delimiter determines the extent or the vicinity till which the negation word will have polarity reversal effect. If we observe a delimiter, we finalize that as our effect scope. Some delimiters include Coordinating conjunction words and Wh-determiners/pronouns. Examples of delimiters are "when," "whenever," "whether," "because," "unless," "until," "since," and "hence."

4 Experimentation and Results

Suppose a sentence *s* is given, we perform two sub-tasks:

1. Subjectivity classification: This determines whether *s* is a subjective or an objective sentence.
2. Sentence-level sentiment classification: This determines whether it expresses a positive or negative opinion, if *s* is subjective.

Construct-based model features:

- Naive Bayes unigram model.
- Identify POS that provide no sentiment information and those that do:
  Provides No Sentiment Information = [‘CD’, ‘DT’, ‘EX’, ‘FW’, ‘IN’, ‘LS’, ‘NNP’, ‘NNPS’, ‘PDT’, ‘POS’, ‘PRP’, ‘PRP$’, ‘RP’, ‘SYM’, ‘TO’, ‘WP’, ‘WRB’].
  Provides Sentiment Information = [‘JJ’, ‘JJR’, ‘JJS’, ‘NN’, ‘NNS’, ‘RB’, ‘RBR’, ‘RBS’, ‘UH’, ‘VB’, ‘VBD’, ‘VBG’, ‘VBN’, ‘VBP’, ‘VBZ’].
- Identifying negation breakers.
- Delimiters = [‘CC’, ‘WDT’, ‘WP$’].
- Identifying default positive POS: [‘MD’].
- Identifying sentiment incrementors and decrementors:
  e.g.: {‘rising’,’extremely’,’slightly’,’increased’,’increasing}
  {‘reduced’,’compensated’,’lessened’,’trivialised’,’decreased’,’trivialized’}.
- Identifying the negation scope.
  #TextAfterNEG in the form of ordered list of words. (Known patterns were identified).
  #output: whether the condition is followed and if yes, we keep count of the depth of negation from point of occurrence.
  # More than one negation phrase is handled properly.

The snapshot of ordered list is shown in Fig. 1.
The results obtained are shown in the Fig. 2.

Fig. 1 Snippet of ordered list of words

```
suffixstructure = {
                  'RB':
                          {
                              'JJ':{}
                          },
                  'DT':
                          {
                              'JJ':{},
                              'RB':
                                  {
                                      'NN':{}
                                  }
                          },
                  'VB': {},
                  'NN': {},
                  'JJ': {}
```

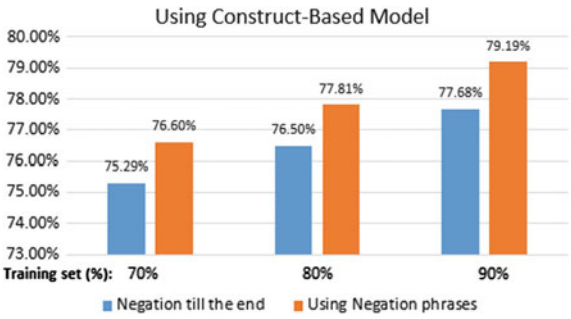

Fig. 2 Result obtained using construct-based model

5 Conclusion and Future Work

The construct-based model, given a set of five constructs (rules), could attain an accuracy of **79.19 %**, so as we expand the domain to more such constructs, the accuracy can then be increased further. Thus, future work would involve incorporating more such composition rules from grammar that affect the sentence polarity.

References

1. Esuli, A., Sebastiani, F.: Sentiwordnet: A publicly available lexical resource for opinion mining. In: Proceedings of LREC. vol. 6, pp. 417–422 (2006).
2. Grefenstette, G., Qu, Y., Shanahan, J. G., Evans, D. A.: Coupling niche browsers and affect analysis for an opinion mining application. In: RIAO. pp. 186–194 (2004).
3. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 168–177. ACM (2004).
4. Huang, J., Lu, J., Ling, C. X.: Comparing naive bayes, decision trees, and svm with auc and accuracy. In: Data Mining, 2003. ICDM 2003. Third IEEE International Conference on. pp. 553–556. IEEE (2003).
5. Neviarouskaya, A., Prendinger, H., Ishizuka, M.: Recognition of affect, judgment, and appreciation in text. In: Proceedings of the 23rd International Conference on Computational Linguistics. pp. 806–814. Association for Computational Linguistics (2010).
6. Niu, Y., Zhu, X., Li, J., Hirst, G.: Analysis of polarity information in medical text. In: AMIA Annual Symposium Proceedings. vol. 2005, p. 570. American Medical Informatics Association (2005).
7. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up?: sentiment classification using machine learning techniques. In: Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10. pp. 79–86. Association for Computational Linguistics (2002).
8. Polanyi, L., Zaenen, A.: Contextual valence shifters. In: Computing attitude and affect in text: Theory and applications, pp. 1–10. Springer (2006).
9. Poria, S., Cambria, E., Winterstein, G., Huang, G. B.: Sentic patterns: Dependency-based rules for concept-level sentiment analysis. Knowledge-Based Systems 69, 45–63 (2014).

10. Sangiorgi, P., Augello, A., Pilato, G.: An approach to detect polarity variation rules for sentiment analysis. In: Proc of WEBIST 2014–10th international conference on web information systems and technologies, 3–5 April (2014).
11. Wilson, T., Wiebe, J., Hoffmann, P.: Recognizing contextual polarity in phrase-level sentiment analysis. In: Proceedings of the conference on human language technology and empirical methods in natural language processing. pp. 347–354. Association for Computational Linguistics (2005).
12. Yu, H., Hatzivassiloglou, V.: Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In: Proceedings of the 2003 conference on Empirical methods in natural language processing. pp. 129–136. Association for Computational Linguistics (2003).