Filtering Malicious Comments with NLTK

Team 23

Authors

20170798 Seungho Kim
20170221 Jaehyeon Myeong
20170305 Hyoungjo Bhang
20170516 Jinwon Lee
20170660 Daeun Choi

Affiliations

School of Computing, KAIST

{skim17, mjhbest, bhanghj3094
, arsenal, cde1103}@kaist.ac.kr

1 Introduction

With the rapid growth of portable technologies and the internet, various platforms are providing different ways of communicating. Among them, comments are notable in its simplicity and robustness, allowing indefinite lengths of communication.

At the same time, however, due to its anonymity, it has become much easier to not only criticize someone but also denounce and attack them with words. Therefore, our work attempts to contribute to creating a ‘better’ cyberspace, adapting NLTK to filter malicious comments.

2 Background/Related Work

User reviews were selected for experiment due to its comparability. We collected over 1 million reviews from Amazon, Mendeley, and Trip Advisor ((Ni et al., 2019), (Benlahbib, 2019), (Wang et al., 2011), (Pang and Lee, 2005), (Blitzer et al., 2007)). Additionally, to evaluate our model with actual data, we crawled comments from YouTube videos.

In order to achieve better results for experiment, we adopted several external models. A lexicon from Vader Sentiment (Hutto and Gilbert, 2014), which maps various words, abbreviations, and text emoticons with a corresponding ”sentiment” score was used in order to replace SentiWordnet, which we found to have a low accuracy, especially with comments. In addition, as a characteristic of internet comments, typos were frequently detected in comment data, so the Autocorrect package was applied for improvement.

3 Approach

Our goal is to distinguish comments that give useful feedback from those that simply reproach others. We built a comment discriminator assessing how positive and negative reviewers are about a content.

To analyze the text, we first analyze each word to extract comparable values in smallest scale. Just as simple this abstraction is, it fails to provide context from other parts of text that can be useful parameters for use. Nearby words, the sentence it is encapsulated in, or even the text as a whole could be used to provide context. Due to the limits of the term-long project, we resided on using nearby words and sentences for context analysis. To make use of this context, we categorized certain features for specific handling.

4 Experiment

The idea of extracting information from review consist of two main aspects. First, we analyze the ‘positivity’ and ‘negativity’ of both individual words and phrases.. The frequency of words, appearances of sayings and memes can fall into this category.

Next, the relationship between words and how they form a text must be dealt by an another method. This method focuses on relationships between words and sentences. Distinctive techniques using part-of-speech tags, the semantics of conjunctions, and giving extra emphasis depending on circumstances would be examined.

4.1 Modes

We chose 9 features from each text to increase the performance: *intensifiers*, *neutralizers*, *uppercase*, *threshold*, *emphasis*, *conjunction*, *exclamation*, *negative words*, and *no/not phrases*.

Intensifiers, and *neutralizers* are labels given to words whose NLTK Synset definitions have more than 2 of 6 keywords ’extent’, ’intensifier’, ’intensity’, ’quantifier’, ’degree’, ’comparative’. Words like ’absolutely’ and ’really’ are considered *intensifiers* while ’rarely’ and ’seldom’ are *neutralizers*.

More importance was given for scores of words and sentences. *Uppercase* and *exclamation* refers to comments such as 'GREAT!!!', in which the extra emphasis are given on specific words. *Emphasis* and *conjunction* amplifies scores for the first or last sentence sentence or the sentence with conjunctions other than stop words like 'however'.

The *negative words* mode simply changes sign of sentiment of the next word if it is negative. Similarly, *no/not phrases* reverse scores of phrases that no or not is modifying and removed the negative sentiment of no/not afterward. Example phrases are "not like" or "disappointed perfectly".

Lastly, *threshold* is applied after all other amplifications, and excludes words whose sentiment score is lower than 1. The weight of importance varies for each mode; 1.4 for words, and 1.5 for sentences as described above. The weights are tuned for best performance between different datasets.

4.2 Scoring Algorithm

The text is tokenized into sentences, then words. The base sentiment is acquired from Vader, and sentiments of phrases like 'dropping the mic' are added as special words. Next, scoring function is run for each sentences with the proposed 9 schemes. The overall score then applies scores and weights from each sentence, according to the modes (*ors*: overall review score, *ss*: sentence score, *ssw*: sentence sentiment weight, *ws*: word score, *wsm*: word sentiment modifiers): $rs = \Sigma ss \times ssw, ss = \Sigma ws \times wsm$

5 Result

To determine accuracy of the algorithm, the original and calculated scores were plotted on the dataset used as training data. Also, We have applied this algorithm to both training and test data. We used Amazon dataset for training, and the TripAdvisor dataset for test. Finally, for our ultimate goal, we applied this algorithm on YouTube to filter out the malicious comments. The result is in Fig 2.

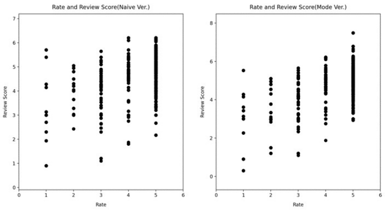
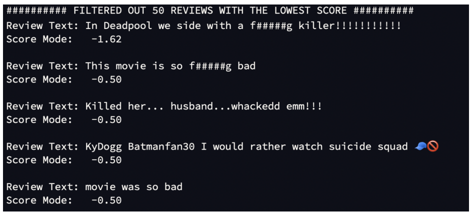


Figure 1: Comparing result with original score

	TP	FP	FN	F-score
Amazon	0.877	0.074	0.049	0.934
TripAdvisor	0.717	0.110	0.174	0.835



```
########## FILTERED OUT 50 REVIEWS WITH THE LOWEST SCORE ##########
Review Text: In Deadpool we side with a f#####g killer!!!!!!!!!!!
Score Mode:   -1.62

Review Text: This movie is so f#####g bad
Score Mode:   -0.50

Review Text: Killed her... husband...whackedd emm!!!
Score Mode:   -0.50

Review Text: KyDogg Batmanfan30 I would rather watch suicide squad 🧢🚫
Score Mode:   -0.50

Review Text: movie was so bad
Score Mode:   -0.50
```

Figure 2: Applying the algorithm on a YouTube video (Censored version)

6 Discussion

The naive results on the left of Fig 1 are extracted by simply calculating the sentiment value of each word, while the results on the right are extracted based on the 9 modes of our algorithm. We can see how the distribution on the right follows a more linear pattern, and therefore the algorithm using mode works more accurately than the one that only judges sentiments.

We also studied about how much the each mode contributed to the better performance, by comparing the results only using one mode. The other cases were all similar, but the uppercase mode showed a better performance. Maybe we can try several combinations of these modes to find a case that shows better performance.

Also we could successfully detect malicious functions from the YouTube comments. Especially by sorting the comments to ascending score, many of them pours out a raw criticism with swear words.

7 Conclusion

There are various ways to potentially improve the quality of the filter. The comments can become positive or negative based on countless criteria, so using previously analyzed big data will be helpful. Also, the language style or pattern differs by platform. Therefore, implementing separate models for each pool of comments may improve the results by dealing with type-based specifications.

Obtaining a discrete value for each comment allows us to compare the results with actual values, or treat them like actual review numbers. All of these potential improvements may help us in various areas; YouTube channels may be able to use this sentiment analyzer for obtaining a numeric value which they may consider as feedback.

References

Abdessamad Benlahbib. 2019. 1000 movie reviews (review + attached rating + sentiment polarity) for reputation generation.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, Bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 440–447, Prague, Czech Republic. Association for Computational Linguistics.

C. Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197, Hong Kong, China. Association for Computational Linguistics.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of ACL*, pages 115–124.

Hongning Wang, Chi Wang, ChengXiang Zhai, and Jiawei Han. 2011. Learning online discussion structures by conditional random fields. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 435–444.