Syntactical Analysis of the Weaknesses of Sentiment Analyzers

Rohil Verma
MIT
77 Massachusetts Avenue
rohil@mit.edu

Samuel Kim
MIT
77 Massachusetts Avenue
sungil@mit.edu

David Walter
MIT
77 Massachusetts Avenue
dwalter@mit.edu

Abstract

We carry out a syntactic analysis of two state-of-the-art sentiment analyzers, Google Cloud Natural Language and Stanford CoreNLP, to assess their classification accuracy on sentences with negative polarity items. We were motivated by the absence of studies investigating sentiment analyzer performance on sentences with polarity items, a common construct in human language. Our analysis focuses on two sentential structures: downward entailment and non-monotone quantifiers; and demonstrates weaknesses of Google Natural Language and CoreNLP in capturing polarity item information. We describe the particular syntactic phenomenon that these analyzers fail to understand that any ideal sentiment analyzer must. We also provide a set of 150 test sentences that any ideal sentiment analyzer must be able to understand.

1 Introduction

Sentiment analysis of texts, from relatively long product and movie reviews (Pang et al., 2002) to short tweets (Go et al., 2009), is a rich and evolving field. Probabilistic analyzers, such as Google's Natural Language client (Google, 2018) and Stanford's CoreNLP package (Manning et al., 2014), have improved in recent years, but major challenges remain in classifying practical sentences. In this paper, we focus on a particular grammatical phenomena analyzers often misclassify: the presence of specific polarity items.

As one of the fundamental components of natural language, polarity items (e.g. nothing, any, ever) are lexical items that can appear only in specific licensing contexts. Thus, we should be able to identify the grammatical polarity of a sentence by the presence of such polarity items, allowing us to determine its sentiment. However, these licensing contexts are challenging to identify and are generally different for each item (Baker, 1970). We aim to understand how negative polarity items are involved in misclassified sentences and use this knowledge to characterize the syntactic phenomenon an ideal sentiment analyzer must learn.

First, we present a brief background on sentiment analyzers and polarity items. In the next section, we describe our methodology in terms of what kinds of sentences we want to use and how we can best test the sentiment analyzers. Our methodology involves trying sentences with negative polarity items under two different licensing contexts, downward entailment and non-monotone quantifiers. We then evaluate variations of the sentences to show that the sentiment analyzers are not correctly using the polarity items. By exploring these misclassified sentences, we describe the particular syntactic configuration that leads to misclassification, presenting weaknesses in the state-of-the-art sentiment analyzers in understanding and handling polarity items.

2 Background and Previous Work

2.1 Sentiment Analyzers

Most sentiment analyzers (Wang et al., 2012; Pak and Paroubek, 2010; Cambria et al., 2013) are based on a statistical approach, relying on a conglomeration of sentiments of the individual words in a sample. The main assumption behind such statistical approaches is that keywords contain essential information to infer the sentiment of a whole sample. Therefore, this type of statistical approach does not readily consider complex syntactic interactions between individual words; instead, the main focus lies in the system's learning of the relevant knowledge through texts relevant to the sentiment analysis task.

Statistical methods often employ bag-of-words as input features and represent a document by the

Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 1122–1127
Brussels, Belgium, October 31 - November 4, 2018. ©2018 Association for Computational Linguistics

summation of all bag-of-words features in that document. A model, such as a maximum entropy classifier or support vector machine (Mullen and Collier, 2004), can be trained to learn which words or combinations of words are relevant for sentiment analysis (Pang et al., 2002). With bag-of-words as the input of a model, we lose spatial structure for a document, so a classifier is incapable of differentiating "I knew the dog would never bite" from "The dog knew the man would never bite" or "Bite never I knew the dog would."

To overcome such challenges, deep learning models have gained popularity for this task. (Glorot et al., 2011) use domain adaptation to train an adversarial network, where two models are pitted against each other: one classifying sentiment and the other creating input documents. This approach allows the system to learn from data sets across multiple domains, increasing the flexibility of the sentiment classifier. However, the weakness of this approach is that the system uses a bigram bag-of-words as input, making it unable to learn long-distance syntactic phenomena.

Recent methods have proposed learning word embeddings (Tang et al., 2014) or applying deep neural architectures (Dos Santos and Gatti de Bayser, 2014) to extract context and sentiment from short texts, as these contain minimal information. Although these techniques have shown performance improvements, they have not been completely successful in capturing long-distance dependencies, leading to the proposal of memory networks (Weston et al., 2014) and attention-based mechanisms (Wang et al., 2016).

2.2 Polarity Items

In Emergence of Meaning (Crain, 2012), Crain puts forward polarity items, like "some" and "any" that are similar, but are sometimes interpreted differently. He presents the example of (1) "John didn't eat any of the kangaroo." and compares it to (2) "John didn't eat some of the kangaroo." The sentence with "some" implies that John did eat a part of the kangaroo, but there is a part of it that he did not eat. The sentence with "any" implies the stronger statement that John did not eat any of the kangaroos. These two interpretations differ due to the polarity of the two words. "Any" is only accepted in negative contexts, so it has negative polarity, whereas "some" can be accepted in both positive and negative contexts and possesses positive polarity. To observe that "any" only works in negative contexts, consider the sentence (3) "John ate any of the kangaroo." which has positive context, and is incorrect with the word "any".

Further, the words "any" and "some" can sometimes be used interchangeably and have the same interpretation. (4) "You'll never convince me that John didn't eat some/any of the kangaroo." contains (1) and (2). In (4) there is negation (never) in a higher clause; then the latter clause contains negation and "any"/"some". These practical examples demonstrate that word ordering in a sentence matters, and that polarity items can exist in complex statements requiring a fundamental understanding of human language for correct interpretation. Therefore, a model just working with bag-of-words or even n-gram features does not appear to be sufficient for practical sentences that require spatial or syntactic understanding.

Polarity items are permitted only within specific licensing contexts, which means they can only occur in specific sentential structures. Our paper explores two licensing contexts: downward entailment and non-monotone quantifiers. Under downward entailment, the sentence acts as a monotone decreasing function such that when parts of the sentence are removed monotonically, the relative strength of a statement monotonically decreases. For example, "nobody moved into the house" implies "nobody moved into the house quickly", so "nobody moved" is a monotone-decreasing phrase. On the other hand, non-monotone quantifiers lack clear downward or upward entailment (Giannakidou, 2002). For example, the phrase "exactly three men never moved" does not entail "exactly three men never moved quickly" and vice-versa, so it has non-monotone entailment. The linguistic phenomenon of licensing contexts and polarity items are a fundamental part of human language, and the metric to measure the performance of a sentiment analyzer should consider how it handles these polarity items.

3 Related Work

To the best of our knowledge, there are no previous studies investigating the weaknesses of Google Natural Language, CoreNLP, or other probabilistic sentiment analyzers in classifying sentences with polarity items.

	Classification accuracy									
	Downward entailment					Non-monotone quantifier				
	a	b	c	d	e	a’	b’	c’	d’	e’
Google NLP	0%	0%	0%	80%	80%	0%	0%	0%	93.3%	86.7%
Stanford CoreNLP	0%	0%	0%	33.3%	53.3%	0%	0%	0%	33.3%	66.7%

Table 1: Summary of classification accuracy on downward entailment and non-monotone quantifiers. Each category indicates 15 test sentences. a: Bill has never done anything [negative adjective], b: [subject phrase] has never done anything [negative adjective], c: Bill has not done anything [negative adjective], d: Bill has done something [negative adjective], e: [subject phrase] has done something [negative adjective]. a’-e’: instead of Bill, we use a non-monotone quantifier (e.g. exactly half).

4 Experiment

4.1 Methodology

We developed 150 test sentences derived from two base sentences, under two different licensing contexts: downward entailment and non-monotone quantifiers. We first tested on Google Cloud Natural Language (Google, 2018) and then repeated the experiment on the Stanford CoreNLP sentiment analyzer (Socher et al., 2013). Each base sentence consists of a subject phrase, a verb phrase, some polarity item(s), and some modifiers (e.g. painful). The rest of the test sentences consist of variations on the base sentences in the specific phrases, polarity items or modifiers used, allowing us to identify the sentence element responsible for misclassification. We chose 15 sentences with minimal sentiment ambiguity per category, allowing us to demonstrate with statistical confidence the inability of these sentiment analyzers to capture syntactic phenomena.

4.2 Results

For both downward entailment and non-monotone quantifiers, we considered two base sentences:

1. (A) has/have [never or not] done anything (B: negative adjectives). These include categories a, b, c, a’, b’ and c’.

2. (A) has/have done something (B: negative adjectives). These include categories d, e, d’, and e’.

For (A), we used subject phrases such as “I” and “Bill and his friends”, and in the case of non-monotone quantifiers, we used subjects such as “exactly half”, “99% of people”, and “exactly ten students”.

4.2.1 Downward entailment

Sentences in category a (eg: “Bill has never done anything terrible”) should be classified as positive. Both Google Natural Language and CoreNLP achieved 0% accuracy on sentences in this category as shown in Table 1. The sentiment analyzers classified these sentences as either negative or neutral. We consider neutral classification incorrect, as the overall sentence is expressing a positive sentiment towards the subject for possessing good moral character. Our varied adjectives reveals that the adjective used is not the reason for misclassfication.

To investigate misclassification further, we varied the subject phrase (A) with fifteen different subject phrases (category b). We observed that sentences such as “Bill has never done anything terrible” and “Bill in my English class has never done anything terrible.” are classified as negative by Google Natural Language and CoreNLP. The magnitude of the negative sentiment remains the same, indicating that minor variations in the subject phrases do not affect the sentiment of the sentence. However, while CoreNLP classifies “Bill has never done anything grumpy” as neutral, it classifies “Sally has never done anything grumpy” as negative. Google Natural Language classifies both as negative, although the score changes slightly. This example demonstrates another weakness of the sentiment analyzers: they treat subject phrases with the same context (“Bill” vs. “Sally”) differently.

Next, we show that the reason for misclassification is not the presence of “never” by varying the sentences to contain “not” (category c; e.g. “Bill has not done anything terrible”). Both Google Natural Language and CoreNLP classify all 15 sentences as negative and yield the same classification accuracy of 0%. We note that for some sentences there is a minor change in the score, but

```
(ROOT
(S
  (NP (NNP Bill))
  (VP (VBZ has)
    (ADVP (RB never))
    (VP (VBN done)
      (S (ADJP (NN anything)
        (JJ terrible)))))
  (.)))
```

Figure 1: Parse tree for “Bill has never done anything terrible.”

```
(ROOT
(S
(ADVP (RB Exactly))
  (NP (NN half))
  (VP (VBD had)
    (ADVP (RB never))
    (VP (VBN done)
      (S
        (ADJP (NN anything)
          (JJ terrible)))))
  (.)))
```

Figure 2: Parse tree for “Exactly half had never done anything terrible.”

the sentiment of sentences remains the same.

We now consider sentences in category (2) that are semantically opposite to sentences in category (1). For example, “Bill has done something terrible” or other variations (types d and e) should be unambiguously classified as negative. Here, Google Natural Language achieves a classification accuracy of 80% (12 out of 15 test sentences) compared to CoreNLP’s 33.3% (5 out of 15 test sentences). For Google Natural Language, it is evident that the element that causes misclassification is the negative polarity item that the analyzer fails to interpret. The analyzer fails to invert the negative sentiment of negative adjectives (B) and thus classifies the sentence as negative or neutral. CoreNLP, however, does poorly on all categories, although it does significantly better on sentences in category (2).

We now consider the parse tree of the original sentence “Bill has never done anything terrible” (Figure 1) to demonstrate the successful parse of “has never done anything terrible” as a single verb phrase. The failure of the analyzers to correctly determine sentiment implies that despite noting the sentence’s hierarchical structure, they are unable to understand long-term dependencies - they lack an understanding of the c-command relation (Radford, 2004).

4.2.2 Non-monotone quantifiers

We repeated the same analysis for non-monotone quantifiers by including non-monotone quantifiers within the subject phrase. For example, “Exactly half had never done anything terrible” and its sister sentences (e.g. “Ninety percent of parents had never done anything terrible”) should be classified as positive. Similar to our findings on downward entailment, both Google Natural Language and CoreNLP achieve a classification accuracy of 0% on the three classes of sentences within category (1): a total of 45 sentences from class (a’), (b’), and (c’) as shown in Table 1. Analyzers again determine “had never done anything terrible” as a single verb phrase, as shown in a parse tree in Figure 2.

For semantically opposite sentences (class (d’) and (e’)), we get a similar result as in downward entailing sentences, although the classification accuracy slightly increases for both analyzers. Again, we consider neutral classification as a misclassification, as the sentence clearly contains either a positive or negative sentiment towards the subject.

We interpret our experiment with non-monotone quantifiers as follows: (1) both CoreNLP and Google Natural Language lack an understanding of the negative polarity item “anything” and (2) the variation of subject phrases results in a minor change in the sentiment. We note that, as in the case of downward entailment, the failure of the analyzers implies an inability to understand the c-command relation (Radford, 2004). We note a limitation of our experiment: the selection of arbitrary adjectives. However, the 15 adjectives used are commonly used to describe humans, and the result was quite consistent. Further, we found coherent results for both downward entailment and non-monotone quantifiers and were able to highlight the lack of long-distance dependency understanding in state-of-the-art analyzers. Finally, we highlight the importance of assessing performance of sentiment analyzers using practical sentences that involve not only negation as in previous studies (Socher et al., 2013), but also polarity items such as the 150 sentences used in our experiment.

5 Discussion

Our experiment indicates that the Stanford CoreNLP sentiment analyzer (Socher et al., 2013) and Google Cloud Natural Language (Google, 2018) do not understand the c-command relation (Radford, 2004). We now argue that this issue is not a result of the training set used, but rather a fundamental inability of the model class. First, we note that long-term dependencies are a well-known weakness of probabilistic models, even those specifically designed to capture them, such as bidirectional RNNs or LSTMs (Zhang et al., 2018). Next, we note that the sentences that we tested are general sentences whose components could be found in any linguistic corpus; in fact, CoreNLP's treebank does contain annotated sentences containing the "... has never ..." or "... has no ..." constructions, and so the model should correctly analyze such sentences. As a result, we draw the conclusion that these models lack an understanding of long-term dependencies, and based on our experiments, they specifically fail to understand the c-command relation (Radford, 2004). Finally, we remind readers that the c-command relation is associated with Chomskyan grammars; and need not be necessary within other models of syntax. Consequently, learning the c-command relation may lead to better analyzers, but it need not be the only way to improve performance.

6 Conclusions

We evaluated two state-of-the-art sentiment analyzers, Stanford CoreNLP (Socher et al., 2013) and Google Cloud Natural Language (Google, 2018), using sentences with negative polarity items under two different licensing contexts: downward entailment and non-monotone quantifiers. Through such analysis, we noted that current analyzers lack a complete understanding of negative polarity items, and by extension, the c-command relation. We have also produced a set of sentences that can be used to test future analyzers. This work can be extended to validate other analyzers, test non-probabilistic sentiment analyzers or build, new improved sentiment analyzers. We have made the set of misclassified sentences available as supplementary material.

7 Acknowledgements

The authors would like to thank Professor Robert C. Berwick for his guidance in preparing this work.

References

C Lee Baker. 1970. Double negatives. *Linguistic inquiry*, 1(2):169–186.

Erik Cambria, Bjorn Schuller, Bing Liu, Haixun Wang, and Catherine Havasi. 2013. Statistical approaches to concept-level sentiment analysis. *IEEE Intelligent Systems*, 28(3):6–9.

Stephen Crain. 2012. *The emergence of meaning*, volume 135. Cambridge University Press.

Cicero Dos Santos and Maira Gatti de Bayser. 2014. Deep convolutional neural networks for sentiment analysis of short texts. In *International Conference on Computational Linguistics*.

Anastasia Giannakidou. 2002. Licensing and sensitivity in polarity items: from downward entailment to nonveridicality. *CLS*, 38:29–53.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 513–520.

Alec Go, Lei Huang, and Richa Bhayani. 2009. Twitter sentiment analysis. *Entropy*, 17:252.

Google. 2018. Google Cloud Natural Language. https://cloud.google.com/natural-language/. [Online; accessed 22-May-2018].

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

Tony Mullen and Nigel Collier. 2004. Sentiment analysis using support vector machines with diverse information sources. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*.

Alexander Pak and Patrick Paroubek. 2010. Twitter as a corpus for sentiment analysis and opinion mining. In *LREc*, volume 10.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics.

A Radford. 2004. English syntax: An introduction.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014. Learning sentiment-specific word embedding for twitter sentiment classification. In *52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014 - Proceedings of the Conference*, volume 1, pages 1555–1565.

Hao Wang, Dogan Can, Abe Kazemzadeh, François Bar, and Shrikanth Narayanan. 2012. A system for real-time twitter sentiment analysis of 2012 us presidential election cycle. In *Proceedings of the ACL 2012 System Demonstrations*, pages 115–120. Association for Computational Linguistics.

Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. 2016. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 606–615.

Jason Weston, Sumit Chopra, and Antoine Bordes. 2014. Memory networks. *CoRR*, abs/1410.3916.

Lei Zhang, Shuai Wang, and Bing Liu. 2018. Deep learning for sentiment analysis : A survey.