Journal of Big Data
a SpringerOpen Journal

METHODOLOGY Open Access

Sentiment analysis using product review data

Xing Fang* and Justin Zhan

*Correspondence:
xfang@aggies.ncat.edu
Department of Computer Science,
North Carolina A&T State University,
Greensboro, NC, USA

Abstract

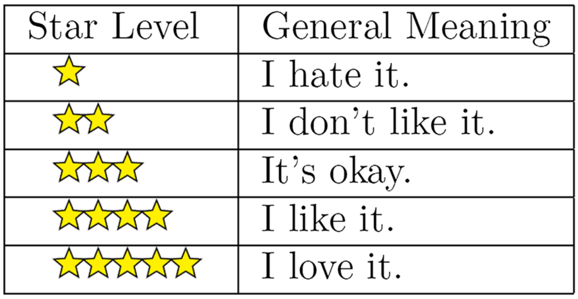
Sentiment analysis or opinion mining is one of the major tasks of NLP (Natural Language Processing). Sentiment analysis has gain much attention in recent years. In this paper, we aim to tackle the problem of sentiment polarity categorization, which is one of the fundamental problems of sentiment analysis. A general process for sentiment polarity categorization is proposed with detailed process descriptions. Data used in this study are online product reviews collected from Amazon.com. Experiments for both sentence-level categorization and review-level categorization are performed with promising outcomes. At last, we also give insight into our future work on sentiment analysis.

Keywords: Sentiment analysis; Sentiment polarity categorization; Natural language processing; Product reviews

Introduction

Sentiment is an attitude, thought, or judgment prompted by feeling. Sentiment analysis [1-8], which is also known as opinion mining, studies people's sentiments towards certain entities. Internet is a resourceful place with respect to sentiment information. From a user's perspective, people are able to post their own content through various social media, such as forums, micro-blogs, or online social networking sites. From a researcher's perspective, many social media sites release their application programming interfaces (APIs), prompting data collection and analysis by researchers and developers. For instance, Twitter currently has three different versions of APIs available [9], namely the REST API, the Search API, and the Streaming API. With the REST API, developers are able to gather status data and user information; the Search API allows developers to query specific Twitter content, whereas the Streaming API is able to collect Twitter content in realtime. Moreover, developers can mix those APIs to create their own applications. Hence, sentiment analysis seems having a strong fundament with the support of massive online data.

However, those types of online data have several flaws that potentially hinder the process of sentiment analysis. The first flaw is that since people can freely post their own content, the quality of their opinions cannot be guaranteed. For example, instead of sharing topic-related opinions, online spammers post spam on forums. Some spam are meaningless at all, while others have irrelevant opinions also known as fake opinions [10-12]. The second flaw is that ground truth of such online data is not always available. A ground truth is more like a tag of a certain opinion, indicating whether the opinion is positive, negative, or neutral. The Stanford Sentiment 140 Tweet Corpus [13] is one of the



Star Level	General Meaning
☆	I hate it.
☆☆	I don't like it.
☆☆☆	It's okay.
☆☆☆☆	I like it.
☆☆☆☆☆	I love it.

Figure 1 Rating System for Amazon.com.

datasets that has ground truth and is also public available. The corpus contains 1.6 million machine-tagged Twitter messages. Each message is tagged based on the emoticons (☺as positive, ☹as negative) discovered inside the message.

Data used in this paper is a set of product reviews collected from Amazon [14], between February and April, 2014. The aforementioned flaws have been somewhat overcome in the following two ways: First, each product review receives inspections before it can be posted [a]. Second, each review must have a rating on it that can be used as the ground truth. The rating is based on a star-scaled system, where the highest rating has 5 stars and the lowest rating has only 1 star (Figure 1).

This paper tackles a fundamental problem of sentiment analysis, namely sentiment polarity categorization [15-21]. Figure 2 is a flowchart that depicts our proposed process for categorization as well as the outline of this paper. Our contributions mainly fall into Phase 2 and 3. In Phase 2: 1) An algorithm is proposed and implemented for negation phrases identification; 2) A mathematical approach is proposed for sentiment score computation; 3) A feature vector generation method is presented for sentiment polarity categorization. In Phase 3: 1) Two sentiment polarity categorization experiments are respectively performed based on sentence level and review level; 2) Performance of three classification models are evaluated and compared based on their experimental results.

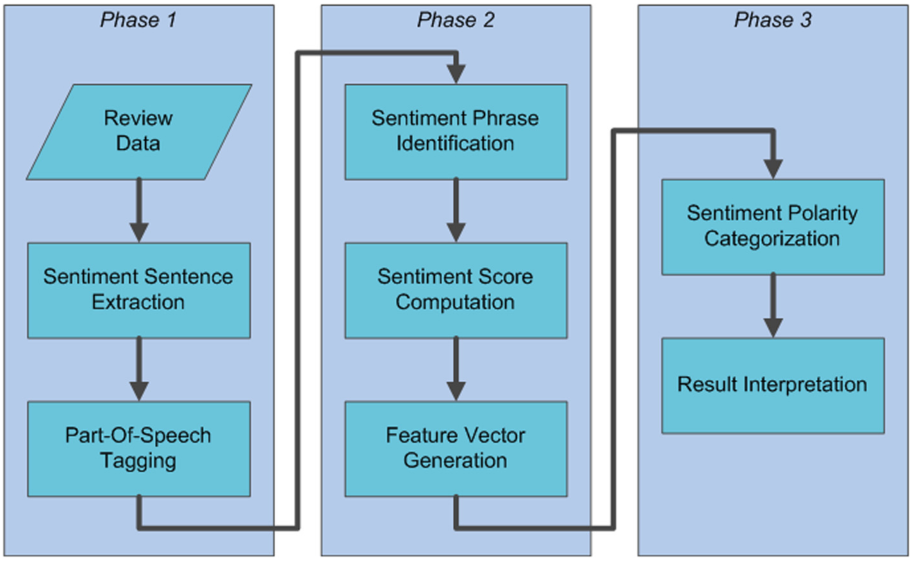

Phase 1
Review Data
Sentiment Sentence Extraction
Part-Of-Speech Tagging
Phase 2
Sentiment Phrase Identification
Sentiment Score Computation
Feature Vector Generation
Phase 3
Sentiment Polarity Categorization
Result Interpretation

Figure 2 Sentiment Polarity Categorization Process.

The rest of this paper is organized as follows: In section 'Background and literature review', we provide a brief review towards some related work on sentiment analysis. Software package and classification models used in this study are presented in section 'Methods'. Our detailed approaches for sentiment analysis are proposed in section 'Background and literature review'. Experimental results are presented in section 'Results and discussion'. Discussion and future work is presented in section 'Review-level categorization'. Section 'Conclusion' concludes the paper.

Background and literature review

One fundamental problem in sentiment analysis is categorization of sentiment polarity [6,22-25]. Given a piece of written text, the problem is to categorize the text into one specific sentiment polarity, positive or negative (or neutral). Based on the scope of the text, there are three levels of sentiment polarity categorization, namely the document level, the sentence level, and the entity and aspect level [26]. The document level concerns whether a document, as a whole, expresses negative or positive sentiment, while the sentence level deals with each sentence's sentiment categorization; The entity and aspect level then targets on what exactly people like or dislike from their opinions.

Since reviews of much work on sentiment analysis have already been included in [26], in this section, we will only review some previous work, upon which our research is essentially based. Hu and Liu [27] summarized a list of positive words and a list of negative words, respectively, based on customer reviews. The positive list contains 2006 words and the negative list has 4783 words. Both lists also include some misspelled words that are frequently present in social media content. Sentiment categorization is essentially a classification problem, where features that contain opinions or sentiment information should be identified before the classification. For feature selection, Pang and Lee [5] suggested to remove objective sentences by extracting subjective ones. They proposed a text-categorization technique that is able to identify subjective content using minimum cut. Gann et al. [28] selected 6,799 tokens based on Twitter data, where each token is assigned a sentiment score, namely TSI(Total Sentiment Index), featuring itself as a positive token or a negative token. Specifically, a TSI for a certain token is computed as:

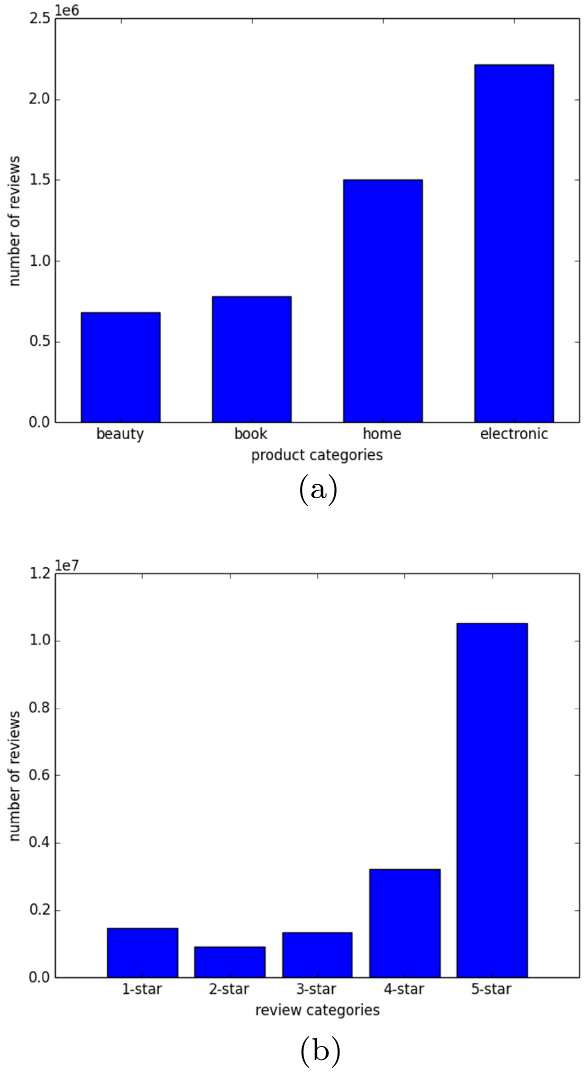
$$TSI = \frac{p - \frac{tp}{tn} \times n}{p + \frac{tp}{tn} * n} \tag{1}$$

where p is the number of times a token appears in positive tweets and n is the number of times a token appears in negative tweets. $\frac{tp}{tn}$ is the ratio of total number of positive tweets over total number of negative tweets.

Research design and methdology

Data collection

Data used in this paper is a set of product reviews collected from amazon.com. From February to April 2014, we collected, in total, over 5.1 millions of product reviews[b] in which the products belong to 4 major categories: beauty, book, electronic, and home (Figure 3(a)). Those online reviews were posted by over 3.2 millions of reviewers (customers) towards 20,062 products. Each review includes the following information: 1) reviewer ID; 2) product ID; 3) rating; 4) time of the review; 5) helpfulness; 6) review text.



(a)

(b)

Figure 3 Data collection **(a)** Data based on product categories **(b)** Data based on review categories.

Every rating is based on a 5-star scale(Figure 3(b)), resulting all the ratings to be ranged from 1-star to 5-star with no existence of a half-star or a quarter-star.

Sentiment sentences extraction and POS tagging

It is suggested by Pang and Lee [5] that all objective content should be removed for sentiment analysis. Instead of removing objective content, in our study, all subjective content was extracted for future analysis. The subjective content consists of all sentiment sentences. A sentiment sentence is the one that contains, at least, one positive or negative word. All of the sentences were firstly tokenized into separated English words.

Every word of a sentence has its syntactic role that defines how the word is used. The syntactic roles are also known as the parts of speech. There are 8 parts of speech in English: the verb, the noun, the pronoun, the adjective, the adverb, the preposition, the conjunction, and the interjection. In natural language processing, part-of-speech (POS) taggers [29-31] have been developed to classify words based on their parts of speech. For sentiment analysis, a POS tagger is very useful because of the following two reasons: 1) Words like nouns and pronouns usually do not contain any sentiment. It is able to

Table 1 Part-of-Speech tags for verbs

Tag	Definition
VB	base form
VBP	present tense, not 3rd person singular
VBZ	present tense, 3rd person singular
VBD	past tense
VBG	present participle
VBN	past participle

filter out such words with the help of a POS tagger; 2) A POS tagger can also be used to distinguish words that can be used in different parts of speech. For instance, as a verb, “enhanced" may conduct different amount of sentiment as being of an adjective. The POS tagger used for this research is a max-entropy POS tagger developed for the Penn Treebank Project [31]. The tagger is able to provide 46 different tags indicating that it can identify more detailed syntactic roles than only 8. As an example, Table 1 is a list of all tags for verbs that has been included in the POS tagger.

Each sentence was then tagged using the POS tagger. Given the enormous amount of sentences, a Python program that is able to run in parallel was written in order to improve the speed of tagging. As a result, there are over 25 million adjectives, over 22 million adverbs, and over 56 million verbs tagged out of all the sentiment sentences, because adjectives, adverbs, and verbs are words that mainly convey sentiment.

Negation phrases identification

Words such as adjectives and verbs are able to convey opposite sentiment with the help of negative prefixes. For instance, consider the following sentence that was found in an electronic device’s review: “The built in speaker also has its uses but so far nothing revolutionary." The word, “revolutionary" is a positive word according to the list in [27].

Algorithm 1 Negation Phrases Identification

```
Require: Tagged Sentences, Negative Prefixes
Ensure: NOA Phrases, NOV Phrases
for every Tagged Sentences do
    for $i/i+1$ as every word/tag pair do
        if $i+1$ is a Negative Prefix then
            if there is an adjective tag or a verb tag in next pair then
                NOA Phrases ← $(i, i+2)$
                NOV Phrases ← $(i, i+2)$
            else
                if there is an adjective tag or a verb tag in the pair after next then
                    NOA Phrases ← $(i, i+2, i+4)$
                    NOV Phrases ← $(i, i+2, i+4)$
                end if
            end if
        end if
    end for
end for
return NOA Phrases, NOV Phrases
```

Table 2 Top 10 sentiment phrases based on occurrence

Phrase	Type	Occurrence
not worth	NOA	26329
not go wrong	NOA	15446
not bad	NOA	15122
not be happier	NOA	14892
not good	NOA	12919
don't like	NOV	42525
didn't work	NOV	38287
didn't like	NOV	21806
don't work	NOV	10671
don't recommend	NOV	9670

However, the phrase "nothing revolutionary" gives more or less negative feelings. Therefore, it is crucial to identify such phrases. In this work, there are two types of phrases have been identified, namely negation-of-adjective (NOA) and negation-of-verb (NOV).

Most common negative prefixes such as not, no, or nothing are treated as adverbs by the POS tagger. Hence, we propose Algorithm 1 for the phrases identification. The algorithm was able to identify 21,586 different phrases with total occurrence of over 0.68 million, each of which has a negative prefix. Table 2 lists top 5 NOA and NOV phrases based on occurrence, respectively.

Sentiment score computation for sentiment tokens

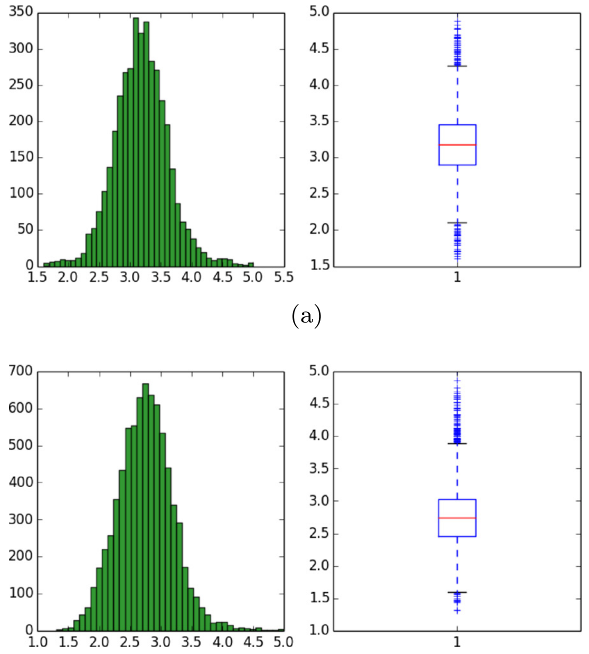
A sentiment token is a word or a phrase that conveys sentiment. Given those sentiment words proposed in [27], a word token consists of a positive (negative) word and its part-of-speech tag. In total, we selected 11,478 word tokens with each of them that occurs at least 30 times throughout the dataset. For phrase tokens, 3,023 phrases were selected of the 21,586 identified sentiment phrases, which each of the 3,023 phrases also has an occurrence that is no less than 30. Given a token t, the formula for t's sentiment score (SS) computation is given as:

$$SS(t) = \frac{\sum_{i=1}^{5} i \times \gamma_{5,i} \times Occurrence_i(t)}{\sum_{i=1}^{5} \gamma_{5,i} \times Occurrence_i(t)} \tag{2}$$

$Occurrence_i(t)$ is t's number of occurrence in i-star reviews, where $i = 1, ..., 5$. According to Figure 3, our dataset is not balanced indicating that different number of reviews were collected for each star level. Since 5-star reviews take a majority amount through the entire dataset, we hereby introduce a ratio, $\gamma_{5,i}$, which is defined as:

$$\gamma_{5,i} = \frac{|5-star|}{|i\text{-}star|} \tag{3}$$

In equation 3, the numerator is the number of 5-star reviews and the denominator is the number of i-star reviews, where $i = 1, ..., 5$. Therefore, if the dataset were balanced, $\gamma_{5,i}$ would be set to 1 for every i. Consequently, every sentiment score should fall into the interval of [1,5]. For positive word tokens, we expect that the median of their sentiment scores should exceed 3, which is the point of being neutral according to Figure 1. For negative word tokens, it is to expect that the median should be less than 3.



(a)

(b)

Figure 4 Sentiment score information for word tokens **(a)** Positive word tokens **(b)** Negative word tokens.

As a result, the sentiment score information for positive word tokens is showing in Figure 4(a). The histogram chart describes the distribution of scores while the box-plot chart shows that the median is above 3. Similarly, the box-plot chart in Figure 4(b) shows that the median of sentiment scores for negative word tokens is lower than 3. In fact, both the mean and the median of positive word tokens do exceed 3, and both values are lower than 3, for negative word tokens (Table 3).

The ground truth labels

The process of sentiment polarity categorization is twofold: sentence-level categorization and review-level categorization. Given a sentence, the goal of sentence-level categorization is to classify it as positive or negative in terms of the sentiment that it conveys. Training data for this categorization process require ground truth tags, indicating the positiveness or negativeness of a given sentence. However, ground truth tagging becomes a really challenging problem, due to the amount of data that we have. Since manually tagging each sentence is infeasible, a machine tagging approach is then adopted as a solution. The approach implements a bag-of-word model that simply counts the appearance of positive or negative (word) tokens for every sentence. If there are more positive tokens than negative ones, the sentence will be tagged as positive, and vice versa. This approach is similar to the one used for tagging the Sentiment 140 Tweet Corpus. Training data for review-level categorization already have ground truth tags, which are the star-scaled ratings.

Table 3 Statistical information for word tokens

Token Type	Mean	Median
Positive Word Token	3.18	3.16
Negative Word Token	2.75	2.71

Feature vector formation

Sentiment tokens and sentiment scores are information extracted from the original dataset. They are also known as features, which will be used for sentiment categorization. In order to train the classifiers, each entry of training data needs to be transformed to a vector that contains those features, namely a feature vector. For the sentence-level (review-level) categorization, a feature vector is formed based on a sentence (review). One challenge is to control each vector's dimensionality. The challenge is actually twofold: Firstly, a vector should not contain an abundant amount (thousands or hundreds) of features or values of a feature, because of the curse of dimensionality [32]; secondly, every vector should have the same number of dimensions, in order to fit the classifiers. This challenge particularly applies to sentiment tokens: On one hand, there are 11,478 word tokens as well as 3,023 phrase tokens; On the other hand, vectors cannot be formed by simply including the tokens appeared in a sentence (or a review), because different sentences (or reviews) tend to have different amount of tokens, leading to the consequence that the generated vectors are in different dimensions.

Since we only concern each sentiment token's appearance inside a sentence or a review,to overcome the challenge, two binary strings are used to represent each token's appearance. One string with 11,478 bits is used for word tokens, while the other one with a bit-length of 3,023 is applied for phrase tokens. For instance, if the *i*th word (phrase) token appears, the word (phrase) string's *i*th bit will be flipped from "0" to "1". Finally, instead of directly saving the flipped strings into a feature vector, a hash value of each string is computed using Python's built-in hash function and is saved. Hence, a sentence-level feature vector totally has four elements: two hash values computed based on the flipped binary strings, an averaged sentiment score, and a ground truth label. Comparatively, one more element is exclusively included in review-level vectors. Given a review, if there are m positive sentences and n negative sentences, the value of the element is computed as: $-1 \times m + 1 \times n$.

Results and discussion

Evaluation methods

Performance of each classification model is estimated base on its averaged F1-score (4):

$$F1_{avg} = \frac{\sum_{i=1}^{n} \frac{2 \times P_i \times R_i}{P_i + R_i}}{n} \tag{4}$$

where P_i is the precision of the *i*th class, R_i is the recall of the *i*th class, and n is the number of classes. P_i and R_i are evaluated using 10-fold cross validation. A 10-fold cross validation is applied as follows: A dataset is partitioned into 10 equal size subsets, each of which consists of 10 positive class vectors and 10 negative class vectors. Of the 10 subsets, a single subset is retained as the validation data for testing the classification model, and the remaining 9 subsets are used as training data. The cross-validation process is then repeated 10 times, with each of the 10 subsets used exactly once as the validation data. The 10 results from the folds are then averaged to produce a single estimation. Since training data are labeled under two classes (positive and negative) for the sentence-level categorization, ROC (Receiver Operating Characteristic) curves are also plotted for a better performance comparison.

Sentence-level categorization

Result on manually-labeled sentences

200 feature vectors are formed based on the 200 manually-labeled sentences. As a result, the classification models show the same level of performance based on their F1-scores, where the three scores all take a same value of 0.85. With the help of the ROC curves (Figure 5), it is clear to see that all three models performed quite well for testing data that have high posterior probability. (A posterior probability of a testing data point, A, is estimated by the classification model as the probability that A will be classified as positive, denoted as $P(+|A)$.) As the probability getting lower, the Naïve Bayesain classifier outperforms the SVM classifier, with a larger area under curve. In general, the Random Forest model performs the best.

Result on machine-labeled sentences

2-million feature vectors (1 million with positive labels and 1 million with negative labels) are generated from 2-million machine-labeled sentences, known as the complete set. Four subsets are obtained from the complete set, with subset A contains 200 vectors, subset B contains 2,000 vectors, subset C contains 20,000 vectors, and subset D contains 200,000 vectors, respectively. The amount of vectors with positive labels equals the amount of vectors with negative labels for every subset. Performance of the classification models is then evaluated based on five different vector sets (four subsets and one complete set, Figure 6).

While the models are getting more training data, their F1 scores are all increasing. The SVM model takes the most significant enhancement from 0.61 to 0.94 as its training data increased from 180 to 1.8 million. The model outperforms the Naïve Bayesain model and becomes the 2nd best classifier, on subset C and the full set. The Random Forest model again performs the best for datasets on all scopes. Figure 7 shows the ROC curves plotted based on the result of the full set.

Review-level categorization

3-million feature vectors are formed for the categorization. Vectors generated from reviews that have at least 4-star ratings are labeled as positive, while vectors labeled as

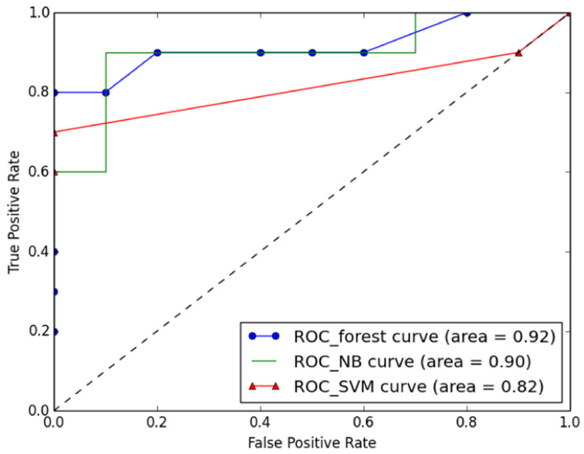


Figure 5 ROC curves based on the manually labeled set.

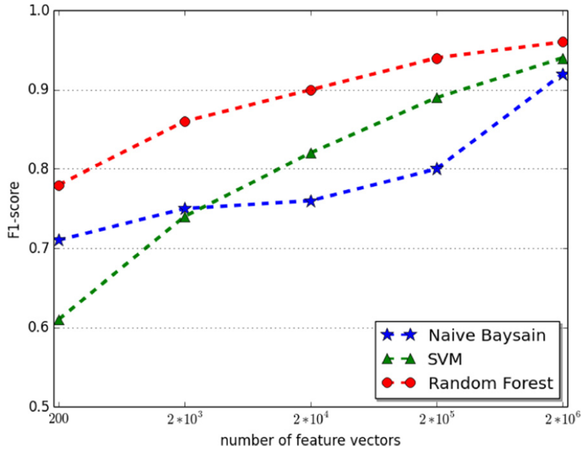


Figure 6 F1 scores of sentence-level categorization.

negative are generated from 1-star and 2-star reviews. 3-star reviews are used to prepare neutral class vectors. As a result, this complete set of vectors are uniformly labeled into three classes, positive, neutral, and negative. In addition, three subsets are obtained from the complete set, with subset A contains 300 vectors, subset B contains 3,000 vectors, subset C contains 30,000 vectors, and subset D contains 300,000 vectors, respectively.

Figure 8 shows the F1 scores obtained on different sizes of vector sets. It can be clearly observed that both the SVM model and the Naïve Bayesain model are identical in terms of their performances. Both models are generally superior than the Random Forest model on all vector sets. However, neither of the models can reach the same level of performance when they are used for sentence-level categorization, due to their relative low performances on neutral class.

The experimental result is promising, both in terms of the sentence-level categorization and the review-level categorization. It was observed that the averaged sentiment score is a strong feature by itself, since it is able to achieve an F1 score over 0.8 for the sentence-level

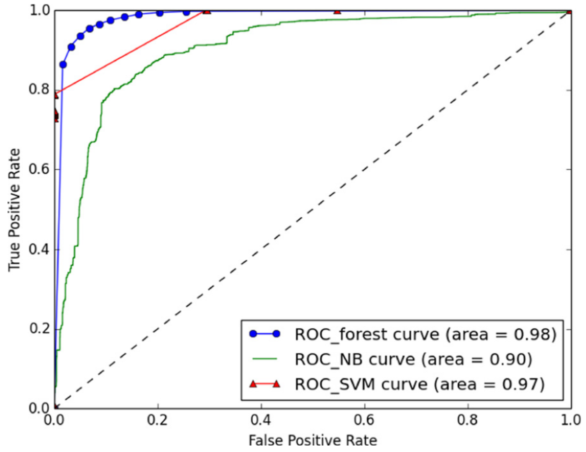


Figure 7 ROC curves based on the complete set.

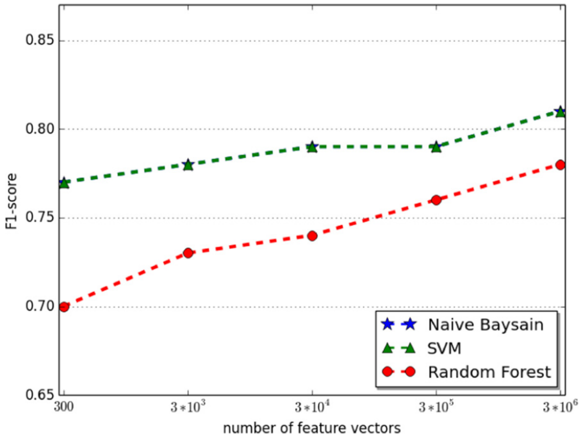


Figure 8 F1 scores of review-level categorization.

categorization with the complete set. For the review-level categorization with the complete set, the feature is capable of producing an F1 score that is over 0.73. However, there are still couple of limitations to this study. The first one is that the review-level categorization becomes difficult if we want to classify reviews to their specific star-scaled ratings. In other words, F1 scores obtained from such experiments are fairly low, with values lower than 0.5. The second limitation is that since our sentiment analysis scheme proposed in this study relies on the occurrence of sentiment tokens, the scheme may not work well for those reviews that purely contain implicit sentiments. An implicit sentiment is usually conveyed through some neutral words, making judgement of its sentiment polarity difficult. For example, sentence like “Item as described.", which frequently appears in positive reviews, consists of only neutral words.

With those limitations in mind, our future work is to focus on solving those issues. Specifically, more features will be extracted and grouped into feature vectors to improve review-level categorizations. For the issue of implicit sentiment analysis, our next step is to be able to detect the existence of such sentiment within the scope of a particular product. More future work includes testing our categorization scheme using other datasets.

Conclusion

Sentiment analysis or opinion mining is a field of study that analyzes people’s sentiments, attitudes, or emotions towards certain entities. This paper tackles a fundamental problem of sentiment analysis, sentiment polarity categorization. Online product reviews from Amazon.com are selected as data used for this study. A sentiment polarity categorization process (Figure 2) has been proposed along with detailed descriptions of each step. Experiments for both sentence-level categorization and review-level categorization have been performed.

Methods

Software used for this study is scikit-learn [33], an open source machine learning software package in Python. The classification models selected for categorization are: Naïve Bayesian, Random Forest, and Support Vector Machine [32].

Naïve Bayesian classifier

The Naïve Bayesian classifier works as follows: Suppose that there exist a set of training data, D, in which each tuple is represented by an n-dimensional feature vector, $X = x_1, x_2, .., x_n$, indicating n measurements made on the tuple from n attributes or features. Assume that there are m classes, $C_1, C_2, ..., C_m$. Given a tuple X, the classifier will predict that X belongs to C_i if and only if: $P(C_i|X) > P(C_j|X)$, where $i, j \in [1, m]\ andi \neq j$. $P(C_i|X)$ is computed as:

$$P(C_i|X) = \prod_{k=1}^{n} P(x_k|C_i) \tag{5}$$

Random forest

The random forest classifier was chosen due to its superior performance over a single decision tree with respect to accuracy. It is essentially an ensemble method based on bagging. The classifier works as follows: Given D, the classifier firstly creates k bootstrap samples of D, with each of the samples denoting as D_i. A D_i has the same number of tuples as D that are sampled with replacement from D. By sampling with replacement, it means that some of the original tuples of D may not be included in D_i, whereas others may occur more than once. The classifier then constructs a decision tree based on each D_i. As a result, a “forest" that consists of k decision trees is formed. To classify an unknown tuple, X, each tree returns its class prediction counting as one vote. The final decision of X's class is assigned to the one that has the most votes.

The decision tree algorithm implemented in scikit-learn is CART (Classification and Regression Trees). CART uses Gini index for its tree induction. For D, the Gini index is computed as:

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2 \tag{6}$$

where p_i is the probability that a tuple in D belongs to class C_i. The Gini index measures the impurity of D. The lower the index value is, the better D was partitioned. For the detailed descriptions of CART, please see [32].

Support vector machine

Support vector machine (SVM) is a method for the classification of both linear and nonlinear data. If the data is linearly separable, the SVM searches for the linear optimal separating hyperplane (the linear kernel), which is a decision boundary that separates data of one class from another. Mathematically, a separating hyperplane can be written as: $W \cdot X + b = 0$, where W is a weight vector and $W = w_1, w2, ..., wn$. X is a training tuple. b is a scalar. In order to optimize the hyperplane, the problem essentially transforms to the minimization of $\|W\|$, which is eventually computed as: $\sum_{i=1}^{n} \alpha_i y_i x_i$, where α_i are numeric parameters, and y_i are labels based on support vectors, X_i. That is: if $y_i = 1$ then $\sum_{i=1}^{n} w_i x_i \geq 1$; if $y_i = -1$ then $\sum_{i=1}^{n} w_i x_i \geq -1$.

If the data is linearly inseparable, the SVM uses nonlinear mapping to transform the data into a higher dimension. It then solve the problem by finding a linear hyperplane.

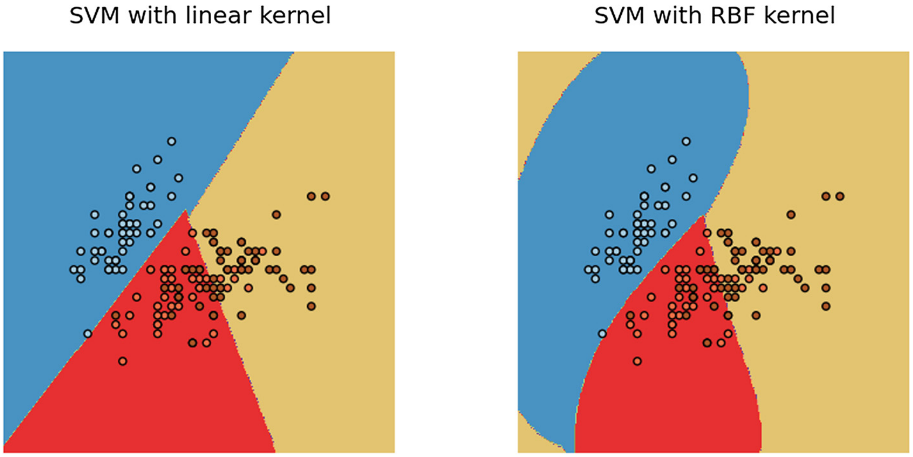


Figure 9 A Classification Example of SVM.

Functions to perform such transformations are called kernel functions. The kernel function selected for our experiment is the Gaussian Radial Basis Function (RBF):

$$K(X_i, X_j) = e^{-\gamma \|X_i - X_j\|^2/2} \tag{7}$$

where X_i are support vectors, X_j are testing tuples , and γ is a free parameter that uses the default value from scikit-learn in our experiment. Figure 9 shows a classification example of SVM based on the linear kernel and the RBF kernel.

Endnotes

[a] Even though there are papers talking about spam on Amazon.com, we still contend that it is a relatively spam-free website in terms of reviews because of the enforcement of its review inspection process.

[b] The product review data used for this work can be downloaded at: http://www.ilabsite.org/?page_id=1091.

Competing interests

The authors declare that they have no competing interests.

Authors' contributions

XF performed the primary literature review, data collection, experiments, and also drafted the manuscript. JZ worked with XF to develop the articles framework and focus. All authors read and approved the final manuscript.

Authors' information

Xing Fang is a Ph.D. candidate at the Department of Computer Science, North Carolina A&T State University. His research interests include social computing, machine learning, and natural language processing. Mr. Fang holds one Master's degree in computer science from North Carolina A&T State University, and one Baccalaureate degree in electronic engineering from Northwestern Polytechnical University, Xi'an, China.
Dr. Justin Zhan is an associate professor at the Department of Computer Science, North Carolina A&T State University. He has previously been a faculty member at Carnegie Mellon University and National Center for the Protection of Financial Infrastructure in Dakota State University. His research interests include Big Data, Information Assurance, Social Computing, and Health Science.

Acknowledgements
This research was partially supported by the following grants: NSF No. 1137443, NSF No. 1247663, NSF No. 1238767, DoD No. W911NF-13-0130, DoD No. W911NF-14-1-0119, and the Data Science Fellowship Award by the National Consortium for Data Science.

Received: 12 January 2015 Accepted: 20 April 2015
Published online: 16 June 2015

References

1. Kim S-M, Hovy E (2004) Determining the sentiment of opinions. In: Proceedings of the 20th international conference on Computational Linguistics, page 1367. Association for Computational Linguistics, Stroudsburg, PA, USA
2. Liu B (2010) Sentiment analysis and subjectivity. In: Handbook of Natural Language Processing, Second Edition. Taylor and Francis Group, Boca
3. Liu B, Hu M, Cheng J (2005) Opinion observer: Analyzing and comparing opinions on the web. In: Proceedings of the 14th International Conference on World Wide Web, WWW '05. ACM, New York, NY, USA. pp 342–351
4. Pak A, Paroubek P (2010) Twitter as a corpus for sentiment analysis and opinion mining. In: Proceedings of the Seventh conference on International Language Resources and Evaluation. European Languages Resources Association, Valletta, Malta
5. Pang B, Lee L (2004) A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In: Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics, ACL '04. Association for Computational Linguistics, Stroudsburg, PA, USA
6. Pang B, Lee L (2008) Opinion mining and sentiment analysis. Found Trends Inf Retr 2(1-2):1–135
7. Turney PD (2002) Thumbs up or thumbs down?: Semantic orientation applied to unsupervised classification of reviews. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL '02. Association for Computational Linguistics, Stroudsburg, PA, USA. pp 417–424
8. Whitelaw C, Garg N, Argamon S (2005) Using appraisal groups for sentiment analysis. In: Proceedings of the 14th ACM International Conference on Information and Knowledge Management, CIKM '05. ACM, New York, NY, USA. pp 625–631
9. Twitter (2014) Twitter apis. https://dev.twitter.com/start
10. Liu B (2014) The science of detecting fake reviews. http://content26.com/blog/bing-liu-the-science-of-detecting-fake-reviews/
11. Jindal N, Liu B (2008) Opinion spam and analysis. In: Proceedings of the 2008 International Conference on, Web Search and Data Mining, WSDM '08. ACM, New York, NY, USA. pp 219–230
12. Mukherjee A, Liu B, Glance N (2012) Spotting fake reviewer groups in consumer reviews. In: Proceedings of the 21st, International Conference on World Wide Web, WWW '12. ACM, New York, NY, USA. pp 191–200
13. Stanford (2014) Sentiment 140. http://www.sentiment140.com/
14. www.amazon.com
15. Go A, Bhayani R, Huang L (2009) Twitter sentiment classification using distant supervision. CS224N Project Report, Stanford. pp 1–12
16. Lin Y, Zhang J, Wang X, Zhou A (2012) An information theoretic approach to sentiment polarity classification. In: Proceedings of the 2Nd Joint WICOW/AIRWeb Workshop on Web Quality, WebQuality '12. ACM, New York, NY, USA. pp 35–40
17. Sarvabhotla K, Pingali P, Varma V (2011) Sentiment classification: a lexical similarity based approach for extracting subjectivity in documents. Inf Retrieval 14(3):337–353
18. Wilson T, Wiebe J, Hoffmann P (2005) Recognizing contextual polarity in phrase-level sentiment analysis. In: Proceedings of the conference on human language technology and empirical methods in natural language processing. Association for Computational Linguistics, Stroudsburg, PA, USA. pp 347–354
19. Yu H, Hatzivassiloglou V (2003) 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. Association for Computational Linguistics, Stroudsburg, PA, USA. pp 129–136
20. Zhang Y, Xiang X, Yin C, Shang L (2013) Parallel sentiment polarity classification method with substring feature reduction. In: Trends and Applications in Knowledge Discovery and Data Mining, volume 7867 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, Heidelberg, Germany. pp 121–132
21. Zhou S, Chen Q, Wang X (2013) Active deep learning method for semi-supervised sentiment classification. Neurocomputing 120(0):536–546. Image Feature Detection and Description
22. Chesley P, Vincent B, Xu L, Srihari RK (2006) Using verbs and adjectives to automatically classify blog sentiment. Training 580(263):233
23. Choi Y, Cardie C (2009) Adapting a polarity lexicon using integer linear programming for domain-specific sentiment classification. In: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2, EMNLP '09. Association for Computational Linguistics, Stroudsburg, PA, USA. pp 590–598
24. Jiang L, Yu M, Zhou M, Liu X, Zhao T (2011) Target-dependent twitter sentiment classification. In: Proceedings of the 49th, Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1. Association for Computational Linguistics, Stroudsburg, PA, USA. pp 151–160
25. Tan LK-W, Na J-C, Theng Y-L, Chang K (2011) Sentence-level sentiment polarity classification using a linguistic approach. In: Digital Libraries: For Cultural Heritage, Knowledge Dissemination, and Future Creation. Springer, Heidelberg, Germany. pp 77–87
26. Liu B (2012) Sentiment Analysis and Opinion Mining. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers
27. Hu M, Liu B (2004) Mining and summarizing customer reviews. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, New York, NY, USA. pp 168–177
28. Gann W-JK, Day J, Zhou S (2014) Twitter analytics for insider trading fraud detection system. In: Proceedings of the sencond ASE international conference on Big Data. ASE
29. Roth D, Zelenko D (1998) Part of speech tagging using a network of linear separators. In: Coling-Acl, The 17th International Conference on Computational Linguistics. pp 1136–1142
30. Kristina T (2003) Stanford log-linear part-of-speech tagger. http://nlp.stanford.edu/software/tagger.shtml
31. Marcus M (1996) Upenn part of speech tagger. http://www.cis.upenn.edu/~treebank/home.html
32. Han J, Kamber M, Pei J (2006) Data Mining: Concepts and Techniques, Second Edition (The Morgan Kaufmann Series in Data Management Systems), 2nd ed. Morgan Kaufmann, San Francisco, CA, USA
33. (2014) Scikit-learn. http://scikit-learn.org/stable/