A Case Study on the Different Algorithms used for Sentiment Analysis

Nikhil George Titus
B.Tech
Department of Computer Science and Engineering
Government Model Engineering college
Thrikkakara Kochi 682023
Kerala, INDIA

Tinto Anto Alapatt
B.Tech
Department of Computer Science and Engineering
Government Model Engineering college
Thrikkakara Kochi 682023
Kerala, INDIA

Niranjan Rao
B.Tech
Department of Computer Science and Engineering
Government Model Engineering college
Thrikkakara Kochi 682023
Kerala, INDIA

ABSTRACT
This paper is a case study on the different algorithms and techniques used to build a robust sentiment analysis system. In this case study we take the problem of classifying movie reviews based on their sentiment. For this, machine learning algorithms based on naive Bayes and support vector machine were used. Naive Bayes with minimum cuts algorithm [1] to generate context sensitive summaries before classifying the document was also used. There are mainly four phases in this process. A tokenizer is used to normalize words and to split the text into sentences. A feature extraction model is used to remove unwanted words and to take into account negation. Then the Naive Bayes with minimum cuts algorithm is used to filter the subjective sentences. Finally, the standard SVM/Naive Bayes algorithm was used to arrive at a proper sentiment.

General Terms
Natural Language Processing, Sentiment Analysis, Algorithms.

Keywords
Tokenizer, Feature Extraction, Classifier, Training, Minimum cuts

1. INTRODUCTION
Text classification is a field in natural language processing where a document is classified into one of the classes based on the training data. Sentiment analysis is a special case of text classification. Sentiment analysis is determining the polarity of a document or text through different algorithms. Sentiment analysis seeks to identify the viewpoints of a text document and classify the document as positive or negative. There are different approaches to solving this problem. Traditional approaches used lexical features like word count to come up with a model. These models tend to be inaccurate and does not adapt well with the context [1]. Hence, machine learning algorithms are the need of the hour. Thus in this paper we analyze the best practices and approaches for solving this problem.

2. PROBLEM STATEMENT
The problem is to classify a given review to positive or negative. A historic data set of reviews which are already tagged as positive/negative is given as input. A huge data set of subjective and objective sentences was also given from these reviews.

3. DETAILED DESIGN
This project has been divided into 7 modules. They are:
1. Tokenizer
2. Feature Extraction
3. Naive Bayes
4. Support Vector Machine
5. Context based subjectivity classifier based on minimum cuts.

Refer Fig 1.1, 1.2 and 1.3 for the detailed high level design.

3.1 Tokenizer
Tokenization is the process of breaking a stream of text into words, phrases, symbols, or other meaningful elements called tokens. The list of tokens becomes input for further processing such as text mining or feature extraction. Tokenization is useful both in linguistics (where it is a form of text segmentation), and in computer science, where it forms part of lexical analysis. It involves dealing with words like cats versus cat. Most of the NLP systems are based on word forms. We need to reduce the words to its single abstract word, or lemma. A lemma is a set of lexical forms having the same stem, the same major part of speech, and the same word sense [5]. Porter's algorithm [2] is a very good algorithm for normalizing words. We use Porter's algorithm for stemming.

3.1.1 Algorithm
1. Read the training corpus and test file.
2. Combine the file in training set.
3. Normalize the input based on porter's algorithm [2].
4. Build a dictionary which is a 2d array of sentence number and sentence text.
5. Write the sentence number with the sentences as a mega document.

3.2 Feature Extraction
The important words can be extracted from the model to improve the accuracy. The features consist of all the adjectives, adverbs and verbs in the document. This is a good method according to the research done by Manish Agarwal and Sudeept Sinha of IIT Karagpur [3].We use a POS tagger to remove unwanted words.

3.2.1 Algorithm

1. Read the mega document generated from the tokenizer.
2. Use Stanford POS tagger to get the verbs, adjectives and adverbs. [4]
3. Use a data structure like trie to remove unwanted words. Use the negation algorithm to handle negations.
4. Write result to another txt file

3.2.2 Negation Algorithm

1. Pass the sentence to split into words function.
2. For each word

2.1. If the word is 'not' or 'nt' or 'n't', then append "NOT" to the beginning of each word after it.

3.3 Context Subjectivity classifier based on minimum cuts.

The minimum cuts algorithm was use to classify the sentence as subjective or objective as suggested by Bo Pang and Lillian Lee [1]. This algorithm makes the model sensitive to context of the document. Here we follow a line by line approach which makes the model very powerful.

3.3.1 Algorithm

1. Accept input from the feature extraction.
2. Train the model based on the minimum cuts algorithm and obtain a list of subjective sentences. [1]

3.4 Naïve Bayes Classifier

Naïve Bayes classifier was used to classify the sentiment to positive or negative. The input is the subjective sentences obtained as an output from the context based subjectivity classifier. Naïve Bayes is a very common supervised probabilistic model. The formula for naïve Bayes is:

$$P\left(\frac{w_k}{C_j}\right) = \frac{n_k + 1}{n + |vocabulary|}$$

Where n is the total number of tokens and P(wk/Cj) is the probability that the word will be in the class and nk= Count of occurrence of wk in the document and P(Cj/d) is the product of all P(Wk/Cj) in test document. Where P(Cj/d) is the probability of the class given the document. [5]

3.4.1 Algorithm

1. Define a class vocabulary containing word, count and probability as its members.
2. Read the document and enter the word and count in it.
3. Calculate probability using naive Bayes.
4. Save this model to an XML file.
5. Apply the model and obtain the probability of the test data to the first class.
6. The test document belongs to one with higher probability.

3.5 Support Vector Machine

Support vector Machines are supervised learning algorithms which are used for classification and regression purposes. The basic SVM model takes a set of input data with labels and constructs a model for classification. This makes it a non-probabilistic linear classifier. SVM constructs a set of hyperplanes which can then be used for classification. Given a set of training examples, SVM builds a model that assigns new example to one of the categories. New examples are marked into the same space and predicted as belonging to one of the categories. For this paper we use Weka an open source machine learning framework to implement SVM [6]. This is an optional model. We can choose one among naïve Bayes or SVM model. If we choose the SVM model the output of the subjectivity classifier is fed into a svm model rather than a Naive Bayes model.

3.5.1 Algorithm

1. Define relationship between attributes using the standard attribute relation file format.(arff)
2. Define arff for train and test data.
3. Train model using arff training data.
4. Save model for future reference using weka.core.SerializationHelper [6]
5. Load test arff data and build classifier using saved model.

4. FUTURE WORK AND CONCLUSIONS

A robust mechanism for developing a very powerful model for analyzing sentiments was described. This approach combines the best approaches in feature engineering and context based classification. By bringing together these approaches we can design a highly powerful sentiment analyzer. The future work includes calculating the quality of the sentiment analyzer. A model has to be implemented and accuracy measured in future. A confusion matrix can be used to define the accuracy of the model generated.

5. ACKNOWLEDGMENT

We would like to thank all who have contributed to our project. A special thanks to Vineetha K.V, Assistant Professor, Department of Computer Science and Engineering. We would also like to thank our friends and family for all their support.

6. REFERENCES

[1] Bo Pang and Lilian Lee: A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts, Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume.

[2] M.F. Porter: An algorithm for suffix stripping, M.F. Porter, (2006) "An algorithm for suffix stripping", Program, Vol. 40 Iss: 3, pp.211 – 218.

[3] Manish Agarwal and Sudeept Sinha: Polarity Detection in Reviews (Sentiment Analysis), CS498: B.Tech. Project Report

[4] Stanford Log-linear POS Tagger http://nlp.stanford.edu/software/tagger.shtml

[5] Daniel Jurafsky, James H Martin: Speech and language processing 2nd edition.

[6] Shilpa Dhanjibhai Serasiya, Neeraj Chaudhary: Simulation of Various Classifications Results using WEKA, International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-1, Issue-3, August 2012

7. APPENDIX

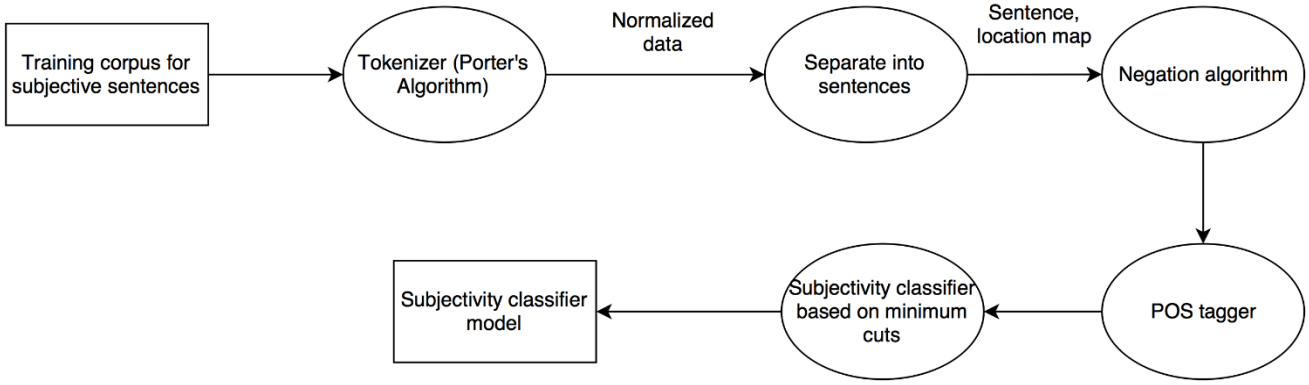

Fig 1.1. Training the subjectivity classifier model

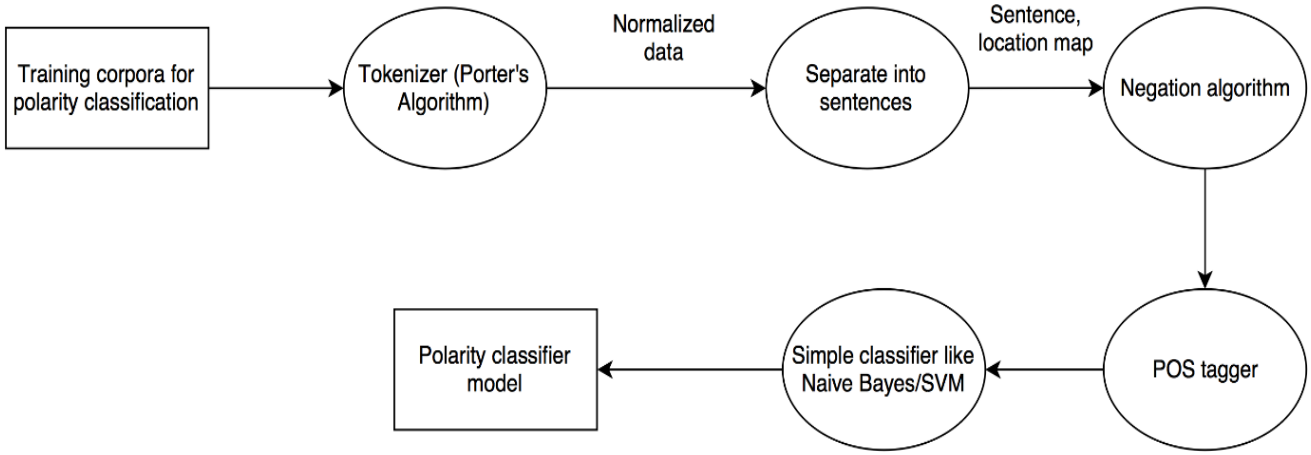

Fig 1.2. Training the polarity classifier model

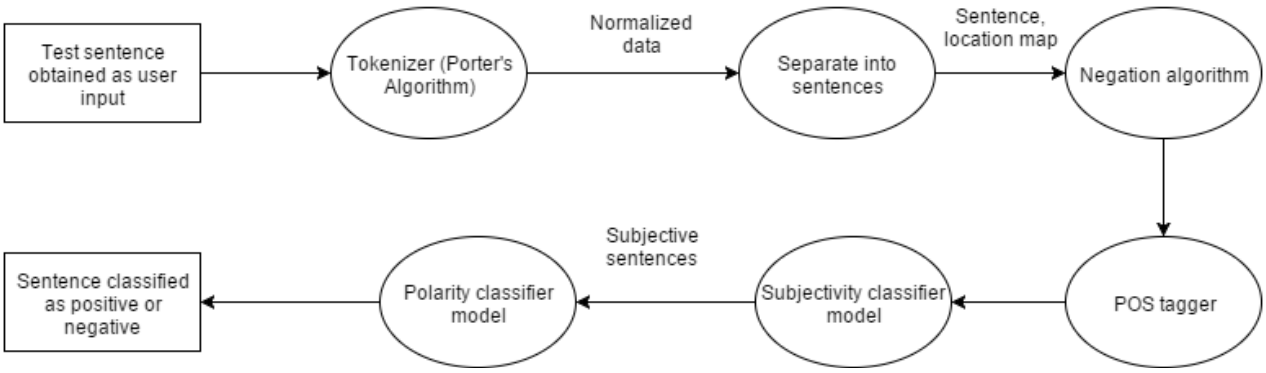

Fig 1.3. Applying the model to a data whose polarity has to be found.