1 Introduction

The Internet is flooded with a huge amount of personal opinion. This ranges from a blog post on the recent U.S. election to a short review for a purchased item on e-commerce website. Understanding what are in the mind of the population may bring a significant monetary or political value, and this gave rise to an active research in *sentiment analysis*, application of natural language processing which aims to mine subjective information.

This report documents an experiment to replicate the work presented in (Wilson et al., 2005) and (Pang et al., 2002). The former takes two symbolic approaches, using a lexicon where each word is associated with a sentiment it represents. The latter was one of the first papers to present a statistical approach to sentiment analysis. Following the description provided in these papers, I build classifiers that take a movie review and judges whether it expresses a positive or negative opinion. The lexicon and the dataset was provided on the framework of the module L90 in MPhil Advanced Computer Science at University of Cambridge. These systems will serve as baselines to the further parts of the task, where I will build my own classifier.

The report proceeds as following. Section 2 describes the approaches presented in (Wilson et al., 2005) and (Pang et al., 2002) in detail. Section 3 describes implementation choices made while setting up the experiment. Finally, Section 4 discusses the results obtained from this experiment.

2 Background

2.1 Symbolic Approach

(Wilson et al., 2005) presents two simple approaches to sentiment analysis using a lexicon which maps a word to *sentiment score*. The sign of the sentiment score represents whether a word is associated with positive or negative document, and the magnitude of the score represents how strong the sentiment is. For instance, a word *disastrous* could have a negative score of high magnitude, and *good* could have a low and positive score. The classification can be done in two ways as shown in Definition 1.

Definition 1 (Symbolic approach) *Let $SLex$ be a lexicon where $SLex[w_i]$ returns the sentiment score associated with w_i. Then functions S_{binary} or $S_{weighted}$ which take a sequence of words are defined as follows.*

$$S_{binary}(w_1...w_n) = \sum_{i=1}^{n} sgn(SLex[w_i])$$

$$S_{weighted}(w_1...w_n) = \sum_{i=1}^{n} SLex[w_i]$$

In the symbolic approach, a sequence of words $w_1...w_n$ can be classified as positive or negative according to the sign of $S_{binary}(w_1...w_n)$ and $S_{weighted}(w_1...w_n)$.

2.2 Naive Bayes Classifier

(Pang et al., 2002) presents how one may use Naive Bayes classifier for sentiment analysis. They frame this task as a classification problem where given a feature vector f that represents a document, $\hat{c}$ among the set of classes C such that

$$\hat{c} = \arg\max_{c \in C} P(c|f)$$

is chosen. Applying Bayes' rule, $P(c|f)$ is

$$P(c|f) = \frac{P(c)P(f|c)}{P(f)}$$

As it is not feasible to compute $P(c|f)$, they make *independence assumption*, which state that each component of f is conditionally independent on class c of the document. Therefore $P(c|f)$ simplifies to

$$P_{NB}(c|f) = \frac{P(c)\prod_{i=1}^{n} P(f_i|c)}{P(f)}$$

When comparing $P_{NB}(c|f)$ for a given f, $P(f)$ is a constant which can be ignored. Therefore the definition of Naive Bayes Classifier used in (Pang et al., 2002) is as follows.

Definition 2 (Naive Bayes Classifier) *Let C be a set of possible classes. The Naive Bayes classifer takes a feature vector f and returns $\hat{c}$ which satisfies the following.*

$$\hat{c} = \arg\max_{c \in C} P(c|f) = \arg\max_{c \in C} P(c) \prod_{i=1}^{n} P(f_i|c)$$

(Pang et al., 2002) computes $P(c)$ and $P(f_i|c)$ using relative-frequency estimation with add-one smoothing from the training set. That is,

$$P(c) = \frac{N_d(c)}{\sum_{c \in C} N_d(c)}$$

$$P(f_i|c) = \frac{N_c(f_i) + \kappa}{N_f(c) + \sum_{w \in V} \kappa}$$

where

- $N_d(c)$ is the number of documents in class c
- $N_c(f_i)$ is the frequency of a feature f_i in all the documents in class c
- $N_f(c)$ is the number of features in all the documents in class c
- κ is the smoothing constant
- V is the set of vocabulary

3 Method

This section describes implementation choices made while replicating the methods described in Section 2.

3.1 Tokenisation

In this section, the process of converting a input documents into a stream of tokens is described. A document is first split into sentences, which get tokenised separately. The first letter of a sentence is capitalised, and it gets split at the white spaces to give the initial list of tokens. Then, 11 rules are applied sequentially to each token in the list. Each rule is associated with a regular expression, and only if the token matches the pattern the rule gets applied. Most rules split at the punctuation which becomes its own token, with the following exceptions.

- Do not split at hyphens if it contains non-alphabetical character. (e.g. "long-term" → ["long", "-", "term"], "3-G21" → ["3-G21"])
- Split "'ll" and add a token "will" instead. (e.g. "you'll" → ["you", "will"])
- Split "n't" and add a token "not" instead. (e.g. "don't" → ["do", "not"])
- Split "'ve" and add a token "have" instead. (e.g. "you've" → ["you", "have"])

Note that all the punctuations are kept in the resulting tokens.

3.2 Cross Validation

I use k-fold cross validation to evaluate and compare the performance of models. The list of all the documents are *randomly* partitioned into k equal-sized fold. The model is tested on each fold after being trained on the remaining $k-1$ folds.

3.3 Symbolic Approach

3.3.1 Sentiment Score

I use the magnitude of 1 for a word marked "strong" and 0.5 for "weak". For words marked "neutral" or "both" I assign the sentiment score of 0.

Table 1: Classification rate

Binary	Weighted	NB
61.9%	64.2%	82.1%

Table 2: Comparison between classifiers

	Binary	Weighted	NB
Binary			
Weighted	==		
NB	>>	>>	

3.4 Naive Bayes Classifier

3.5 Features

I use unigrams without stemming or stoplist. They are admittedly very simple, which I believe is appropriate for a baseline system.

3.6 Smoothing

For the smoothing constant, I use 1 as in (Pang et al., 2002).

4 Result

The table 1 shows the classification rate for each classifier. Note that the classification rate for the Naive Bayes classifier is the mean of 10 results from each fold in 10-fold cross validation. From this result, the Naive Bayes classifier seems to largely outperform the symbolic classifiers. The table 2 shows the result of two-tailed sign test, where == marks $p >= 0.01$, > marks $0.01 > p >= 0.005$ and >> marks $0.005 > p$. From this, it is concluded that the Naive Bayes classifier is significantly better than the symbolic classifiers, whose performance is comparable to each other.

References

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.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 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*, pages 347–354. Association for Computational Linguistics.