Topic Based Sentiment Analysis on Twitter

BY

QUAH SOONG YING

A project report submitted in partial fulfilment of the
requirements for the award of Bachelor of Science (Hons.)
Applied Mathematics with Computing

Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

January 2014

DECLARATION OF ORIGINALITY

I hereby declare that this project report entitled "**TOPIC BASED SENTIMENT ANALYSIS WITH TWITTER**" is my own work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree or award at UTAR or any other institutions.

Signature : _____

Name : _____

ID No. : _____

Date : _____

APPROVAL FOR SUBMISSION

I certify that this project report entitled "**TOPIC BASED SENTIMENT ANALYSIS WITH TWITTER**" was prepared by **QUAH SOONG YING** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Science (Hons.) Applied Mathematics with Computing at Universiti Tunku Abdul Rahman.

Approved by,

Signature : _____

Supervisor : _____

Date : _____

ACKNOWLEDGEMENTS

I would like to appreciate my utmost gratitude to my Final Year Project supervisor, Dr. Tay Yong Haur. You have been a tremendous mentor for me, encouraging and advising both on my research and studies, giving all those useful comments and remarks as well as having discussions so that I could have a better idea for improving and understanding my project.

At the same time, I would also like to thank my family for their continuous support in terms of giving me mental support, motivations, care and everything.

Last but not least, I am thankful to have friends who would help relieve, cheer me up and motivate me when I faced issues.

TOPIC BASED SENTIMENT ANALYSIS WITH TWITTER

QUAH SOONG YING

ABSTRACT

Sentiment analysis is widespread in many countries and also widely used in a lot of application. In this generation that mostly revolves around social networking, applying sentiment analysis on social networks is undoubtedly a hot research area in computer science. In Twitter, popular information that are either facts or opinions are propagated throughout the network. However, unlike normal documents, the restricted length of the messages along with constant changing internet slangs in Twitter, have become a challenge for researchers.

In this project, the analysis of using topic models in tackling the problem is carried out. Topic models are good for exploration of data and helps combat sparse data problem as it assumes that a document is formed out of various topics, Twitter messages expressed by internet users can also be topic-related. Hence, the proposed method is to use Latent Dirichlet Allocation to extract keywords that will be processed into features loaded into Support Vector Machine classifier. The study of the method's quality and effectiveness is also discussed.

The proposed method is implemented and a few sets of features were tested. The implemented model showed that usage of stopwords inside the dataset does hinder the performance of the classifier and that by separating the positive tweets and negative tweets to extract the topics and the keywords, it achieves a better result than combining the positive and negative tweets together. Other than that, it also shows that by choosing k random topics out of a total of t topics, the result yielded is lower.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

LDA	**Latent Dirichlet Allocation**
MaxEnt	**Maximum Entropy**
NB	**Na ïve Bayes**
pLSI	**probabilistic Latent Semantic Indexing**
SO-CAL	**Semantic Orientation Calculator**
STS	**Standard Twitter Sentiment**
SVM	**Support Vector Machine**
URL	**Uniform Resource Locator**

CHAPTER 1: INTRODUCTION

1-1 Background and Motivation

Ever since the rise of social media networking services, there has been an increase of interest in natural language processing community to develop techniques that could mine opinion from texts such as tweets by users. Twitter, as one of the popular and fast-growing social network, can be seen as a platform that contains valuable information, perspectives and opinions of users on issues related to business and society (Jansen et al., 2009). Hence, Twitter is a preferred micro-blogging service used for sentiment analysis. For most of the text analysis, the "bag of words" model is often used. However, it is considered unsatisfactory for sentiment analysis that is topic-dependent in the sense that it does not take into account the relationship between words.

Therefore in the recent years, Topic Modeling has become more popular in some text-related tasks. Topic model is a type of statistical model for discovering topics that occur in a document. It also models the relationship between words. For example, given a document with a topic about education, it is expected words such as "school" and "study" to appear more frequently in the document. Many topic models were proposed by researchers in the past such as latent semantic analysis, latent Dirichlet allocation (D Blei, A Ng, M Jordan, 2002), hierarchical Dirichlet process (HM. Wallach, 2006) and so on.

One of the topic models, latent Dirichlet allocation has been widely used for probabilistic text modeling. Latent Dirichlet allocation (LDA) represents documents as mixtures of latent topics. In comparison with other models, latent Dirichlet allocation consistently does better than the other models in terms of perplexity, classification and collaborative filtering (D Blei, A Ng, M Jordan, 2002). Other than that, Wei and Croft had also shown good performance in Information Retrieval for LDA-based document model.

1-2 Project Objectives

The main objective of this project is to research and evaluate sentiment analysis with Twitter through different methods of text processing, feature extractions as well as classifiers. This project hopes to identify and develop an algorithm that could have a better performance at Twitter sentiment analysis through topic modeling. Tweets by user from Twitter will be used for training the program and testing it for evaluation.

1-3 Project Scope

For this project, only data regarding Twitter tweets that are in English language will be taken into consideration. Classifications made will be based on sentiment analysis in which polarity of each tweets, either positive or negative is determined. As for the process of feature selection, features will be in the form of ‘unigrams’.

CHAPTER 2: LITERATURE REVIEW

Ever since the spike of social media networking services, sentiment analysis on social networks has become a hot topic in the research community.

2-1 Sentiment Analysis

Sentiment analysis is a field of study over application of natural language processing, computational linguistics, and text analytics. In general, sentiment analysis judges the polarities of user opinion towards entities such as products, services, organizations, individuals, events, topics and much more. Sentiment analysis is often conducted at one of three levels: document level, sentence level and feature level - whether the opinion expressed is positive, negative or neutral. The literature survey done has indicated two types of techniques for sentiment analysis, namely, the machine learning approach and lexicon-based approach.

2-1-1 Lexicon based sentiment classification

(Taboada, Brooke, Tofiloski, Voll, & Stede, 2011) Lexicon based sentiment classification uses semantic orientation - a measure of subjectivity and opinion in text. First of all, dictionaries of words annotated with the word's semantic orientation are being used. The calculation of sentiment in SO-CAL (Semantic Orientation Calculator) begins with two assumptions: that individual words that have what is referred to as priority polarity, that is a semantic orientation that is independent of context; and semantic orientation can be represented as a numerical value.

2-1-2 Machine learning sentiment classification

Basically, this approach is like asking the computer program to learn the classification function from a number of labeled documents to then perform classification on unlabeled documents. To do this, the 'bag of word' model is often used as a feature for classification. The 'bag of word' model is a representation of text in an unordered collection of words.

(Pang, Lee, & Vaithyanathan, 2002) who were the first to apply machine learning algorithm into the problem of sentiment categorization considered three machine learning algorithm which are naive Bayes, maximum entropy and support vector machine. Each algorithm are different from one another but each has been shown to be effective in text categorization studies. However, they also stated in their paper that all three machine learning algorithms they employed did not perform as well as sentiment classification based on topic-based categorization.

2-1 Naïve Bayes

Naïve Bayes (NB) is a simple probabilistic model that is based on Bayes' theorem but with a stronger independence assumption. It can be represented by,

$$P_{NB}(c|d) = \frac{P(c)(\prod_{i=1}^{m} P(f_i|c)^{n_i(d)})}{P(d)}$$

where $P(d), P(c)$ is the prior probability of d and c, $n_i(d)$ is the number of times f_i occurs in document d. It is assumed that f_i is conditionally independent to every other feature f_j for $i \neq j$ hence it is represented by $\prod_{i=1}^{m} P(f_i|c)^{n_i(d)})$.

(Domingos & Pazzani, 1997) showed that for problems with highly dependent features, Naïve Bayes would be the optimal classifier but for problems that are more sophisticated, there are other algorithms that outperforms Naïve Bayes.

2-2 Maximum Entropy

Unlike Naïve Bayes model, Maximum Entropy (MaxEnt) does not assume statistical independence of the variables. Rather than that, the model searches for a set of parameter that maximizes the likelihood of the training data. Its estimate of $P(c|d)$ takes the form of:

$$P_{ME}(c|d) \coloneqq \frac{1}{Z(d)} \exp\left(\sum_i \lambda_{i,d} F_{i,c}(d,c)\right)$$

where $Z(d)$ is a normalization function. $F_{i,c}$ is a feature/class function for feature f_i and class c. The $\lambda_{i,d}$ is the feature-weight parameters. The parameters are found using iterative optimization techniques so it may take a long time to learn for large training sets.

2-3 Support Vector Machine

Support vector machine (SVM) has been highly effective and outperforms other classification methods at traditional text categorization. In contrast to NB and MaxEnt, SVM is large-marginal and is based on finding a hyperplane that separates the support vectors which are the data points that lie closest to the decision surface. The solution for the hyperplane vector, $\ddot{w}$ can be given by:

$$\ddot{w} \coloneqq \sum_j \alpha_j c_j \overrightarrow{d_j}, \alpha_j \geq 0$$

where α_j is obtained by solving dual optimization problem, $c_j \in \{1, -1\}$ and $\overrightarrow{d_j}$ is the support vector that contributes most to $\ddot{w}$.

2-2 Topic model

For standard learning algorithms, it is less often for topics-related sentiment analysis to be analyzed. Though there are some work attempting to incorporate sentiment factor into topic models like probabilistic latent semantic indexing (PLSI) or latent Dirichlet allocation (LDA) (Lin & He, 2009). Other than that, (Wallach, 2006) also did graphical comparisons between hierarchical Dirichlet language model, latent Dirichlet allocation and bigram topic model in terms of information rates of test data and mean time taken to perform single iteration of the Gibbs EM algorithm.

2-2-1 Latent Dirichlet allocation

LDA is a generative model that assigns topics to each documents and topic distributions are generated over words given a collection of text. Given the corpus-level parameters α and β, the joint distribution of a topic mixture θ, a set of N topics z, and a set of N words w is given by:

$$p(\theta, z, w|\alpha, \beta) = p(\theta|\alpha)\prod_{n=1}^{N} p(z_n|\theta)\, p(w_n|z_n, \beta)$$

where $p(z_n|\theta)$ is simply θ_i for the unique i such that $z_n^i = 1$. Integrating over θ and summing over z, the marginal distribution of a document is obtained:

$$p(w|\alpha, \beta) = \int p(\theta|\alpha)\left(\prod_{n=1}^{N} p(z_n|\theta)\, p(w_n|z_n, \beta)\right) d\theta$$

Then, taking the product of the marginal probabilities of single documents, the probability of a corpus is given by:

$$p(D|\alpha, \beta) = \prod_{d=1}^{M} \int p(\theta_d|\alpha)\left(\prod_{n=1}^{N_d}\sum_{Z_{dn}} p(Z_{dn}|\theta_d)\, p(w_{dn}|z_{dn}, \beta)\right) d\theta_d$$

The graphical model of LDA is represented in Figure 2.1. As shown in Figure 2.1, LDA's model has three levels namely, the corpus level variables α and β, the document-level variables θ_d and the word-level variables z_{dn} and w_{dn}. Each variables are sampled once in their respective levels.

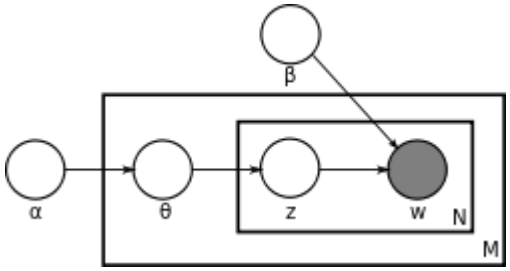

Figure 2.1: Graphical Model of LDA

M denotes the number of documents and N the number of words in the document.

2-2-2 Probabilistic latent semantic indexing

Probabilistic latent semantic indexing (pLSI) also known as probabilistic latent semantic analysis is a statistical technique for the analysis of two-mode and co-occurence data.

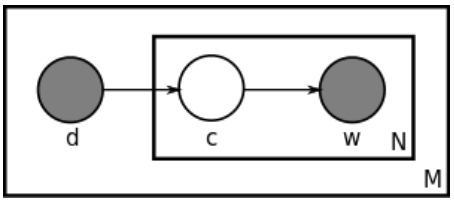

Figure 2.2: Graphical Model of pLSI

The graphical model of pLSI in Figure 2.2 posits that a document label d and a word w_n are conditionally independent given an unobserved topic:

$$p(d, w_n) = p(d) \sum_z p(w_n|z)p(z|d)$$

In contrast to LDA, pLSI assumes that in the mixture of unigrams model, each document is generated from only one topic. Although the model could identify that a document could contain multiple topics, the model can only learn topic mixtures for those documents on which it is trained. Other than that, pLSI risks having the problem of overfitting as there is a linear growth of parameters with respect to the number of documents. Overfitting occurs when a model is having too many parameters relative to the number of observation.

2-3 Evaluation

Evaluation is where the performance of the classification model is determined. For evaluation, a testing set that is distinct from the training set is used in order to prevent any misleading or inaccurate results as the model might memorized all the inputs without actually learning. There are a few metrics that can be used to evaluate the classifier:

3.1 Accuracy

Accuracy measures the percentage of inputs in the test set that the classifier correctly identified. For example, the classifier predicts correctly 30 positive tweets out of 50 tweets. This would give an accuracy of 30/50 = 60%.

3.2 Precision

Precision measures the exactness of the classifier and is defined as $Precision = \frac{Tp}{Tp+Fp}$ where Tp is true positive and Fp is false positive. For example, the classifier predicts 45 positive tweets but only 30 of them are correct. This would give a precision of 30/45 = 67%.

3.3 Recall

Recall measures the sensitivity of the classifier and is defined as $Recall = \frac{Tp}{Tp+Fn}$ where Tp is true positive and Fn is false negative. For example, out of 60 positive tweets, the classifier correctly predicts 40 tweets. This would give a recall of 40/60 = 67%.

3.4 F-measure

F-measure is combination of precision and recall and is the weighted harmonic mean of precision and recall. It is defined as $F = 2 \cdot \frac{Precision \cdot Recall}{Precision+Recall}$

2-4 Challenges and difficulties in Twitter sentiment analysis

Besides the challenges that are normally faced by natural language processing and text processing, there are many other challenges for tweet sentiment analysis:

- Each tweets have a limitation of up to 140 characters. The tweets are very short and may also be incomplete. Compared to most sentiment analysis which is in document level, most tweets are sentence level. Since each tweet is so short and sometimes incomplete that they may lack some important information. An example of such tweet: "I want to be back in LA". This tweet lacks information as we do not know what user is trying to express. The user may have tweeted something related to this tweet that may give us a better idea on what the user is trying to express.

- As human language may use different forms of words to express the same opinion, there may be a lot of language variation where the content of the tweets may involve irony and sarcasm. An example of such tweet: "Hurray!!....This sucks!"

- Tweets contain a lot of noisy texts. They are usually written in ambiguous words and abbreviation words. Most of the tweets are also informal and unedited texts where they may contain a lot of grammatical errors or repetitive words or letters in words.

- There are also difficulties dealing with comparisons. For example, in the phrase "Apple is better than Orange", the tweet would be considered positive for Apple and negative for Orange.

- Handling negations are also difficult to handle properly. As there are many forms of negation including verbs like "lacked", "denied" and adverbs like "hardly", "rarely" as well as "not".

CHAPTER 3: RESEARCH METHODOLOGY

3-1 System Description

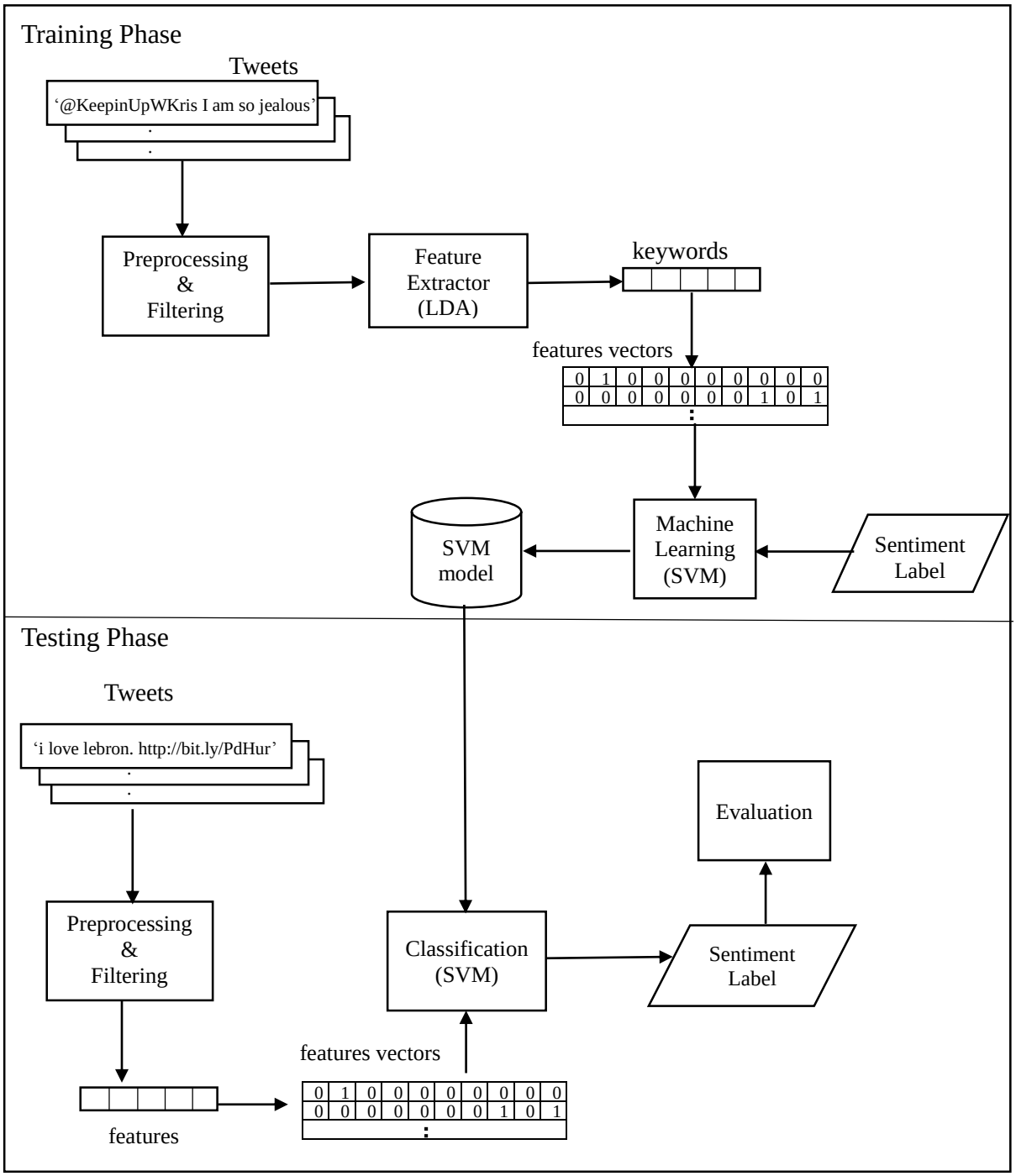

Figure 3.1: System Architecture

The system in Figure 3.1 is designed to evaluate how a topic based sentiment analysis will perform given that the data is taken from Twitter. There are 2 stages for the system which are the training phase and the testing phase. In the training phase, tweets retrieved from Twitter will go through a series of preprocessing and filtering to remove 'noisy' texts such as hashtags, URLs and etc. After that, the processed tweets that are in the form of features are inputted to LDA to find the topics related and the most contributed keywords to the topics. The keywords are compiled to be used as the feature lists, along with the training tweets to define the parameters of the SVM model.

During testing phase, another set of tweets retrieved from Twitter will go through preprocessing and filtering. The feature vectors from preprocessing and filtering are then parsed to the SVM model which has already been trained to predict the label of the tweets. The predicted labels are then collected for evaluations.

3-2 Implementation

3-2-1 Feature Extraction

As we assume that documents are a mixture of topics where there is a probability distribution of topics over words (Blei, et al., 2003), LDA is the chosen method for feature selection as it is a rather simple model for dimensionality reduction and can do summarization of tweets. LDA is simple to use as a module in more complex models (Blei, et al., 2003).

In this project, there are two proposed ways of feature extraction, one of it will be to separate the positive tweets and negative tweets of the training set and then by using LDA, find the latent topics and the most contributing keywords to the topics. Another way would be to combine both positive tweets and negative tweets together into a document and then extract the keywords.

Each document is gone through and each word in the document is randomly assigned to one of the K topics. Then, each word w in document d is gone through. For each topic t, the proportion of words in the document d that is assigned to topic t, $p(t|d)$ and the proportion of assignments to a topic to all documents that come from the word $p(w|t)$ are computed. Then the word is reassigned to a new topic by the probability of the topic that generated word. $p(t|d) \cdot p(w|t)$. After that, the topics with their most contributing keywords learned by LDA model will be collected to be used as features for classification. The tool used is gensim library in Python. (Řehůřek & Sojka, 2010).

```
import os
import logging, gensim, bz2

dir = r'C:\Users\HP\Documents\FYP\Final\training set\separate\withoutstop';
os.chdir(dir);

logging.basicConfig(format='%(asctime)s : %(levelname)s : %(message)s', level=logging.INFO);

# load id->word mapping (the dictionary), one of the results of step 2 above
id2word = gensim.corpora.Dictionary.load_from_text('negative_without.txt');

# load corpus iterator
mm = gensim.corpora.MmCorpus('negative_without.mm');

# extract 30 topics using 20 full passes, with no online updates
lda = gensim.models.ldamodel.LdaModel(corpus=mm, id2word=id2word, num_topics=30, update_every=0, passes=20)
```

Figure 3.2: Code Snippet of Gensim LDA Implementation

As shown in Figure 3.2 are the codes for implementing LDA which was built accordingly to the given template by gensim tutorial. Before running LDA, there is a need to prepare the corpus. There steps shown in Figure 3.3 are the preparation of the corpus for LDA. Firstly, all the words in the documents are assigned a unique id by using a dictionary. Then, the documents are converted to vectors with the function doc2bow() which counts the number of occurrences of each distinct word and returns the results as a sparse vector.

Next on, load the corpus iterator and dictionary into the LDA model. The results from the LDA model can be seen in Figure 3.4 which shows 6 random topics and the 10 most contributing words for each topic. For this project, only 30 topics are taken into account and the 10 most contributing words from the topics are collected as features.

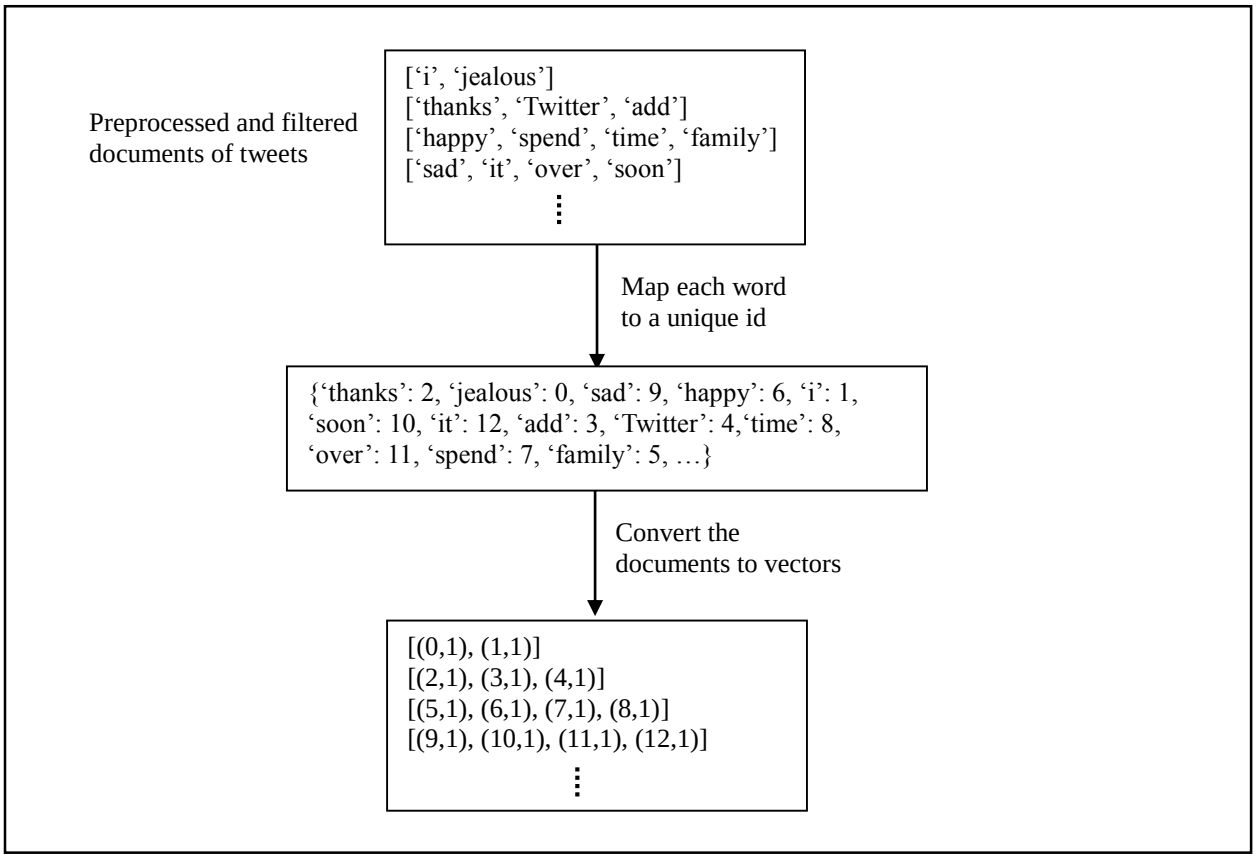

Figure 3.3 Stages of Corpus Preparation for LDA

```
>>> lda.print_topics(30)
topic #0: 0.065*my + 0.031*just + 0.028*the + 0.024*got + 0.021*i + 0.018*from +
 0.017*on + 0.015*her + 0.015*me + 0.014*phone
topic #1: 0.045*to + 0.038*a + 0.028*the + 0.024*with + 0.021*on + 0.015*i + 0.0
15*bored + 0.014*of + 0.013*them + 0.012*and
topic #2: 0.069*day + 0.044*a + 0.037*and + 0.032*the + 0.023*is + 0.023*nice +
0.020*sun + 0.019*in + 0.016*i + 0.014*have
topic #3: 0.053*they + 0.051*are + 0.024*i + 0.020*and + 0.020*as + 0.019*omg +
0.019*the + 0.017*people + 0.015*so + 0.013*cute
topic #4: 0.066*the + 0.040*yeah + 0.039*is + 0.029*of + 0.021*in + 0.019*its +
0.013*that + 0.013*and + 0.013*hot + 0.011*on
topic #5: 0.079*oh + 0.035*sorry + 0.026*a + 0.024*good + 0.018*no + 0.017*luck
+ 0.017*for + 0.016*that + 0.015*thats + 0.014*the
```

Figure 3.4 Example of LDA Topics

3-2-2 Classification

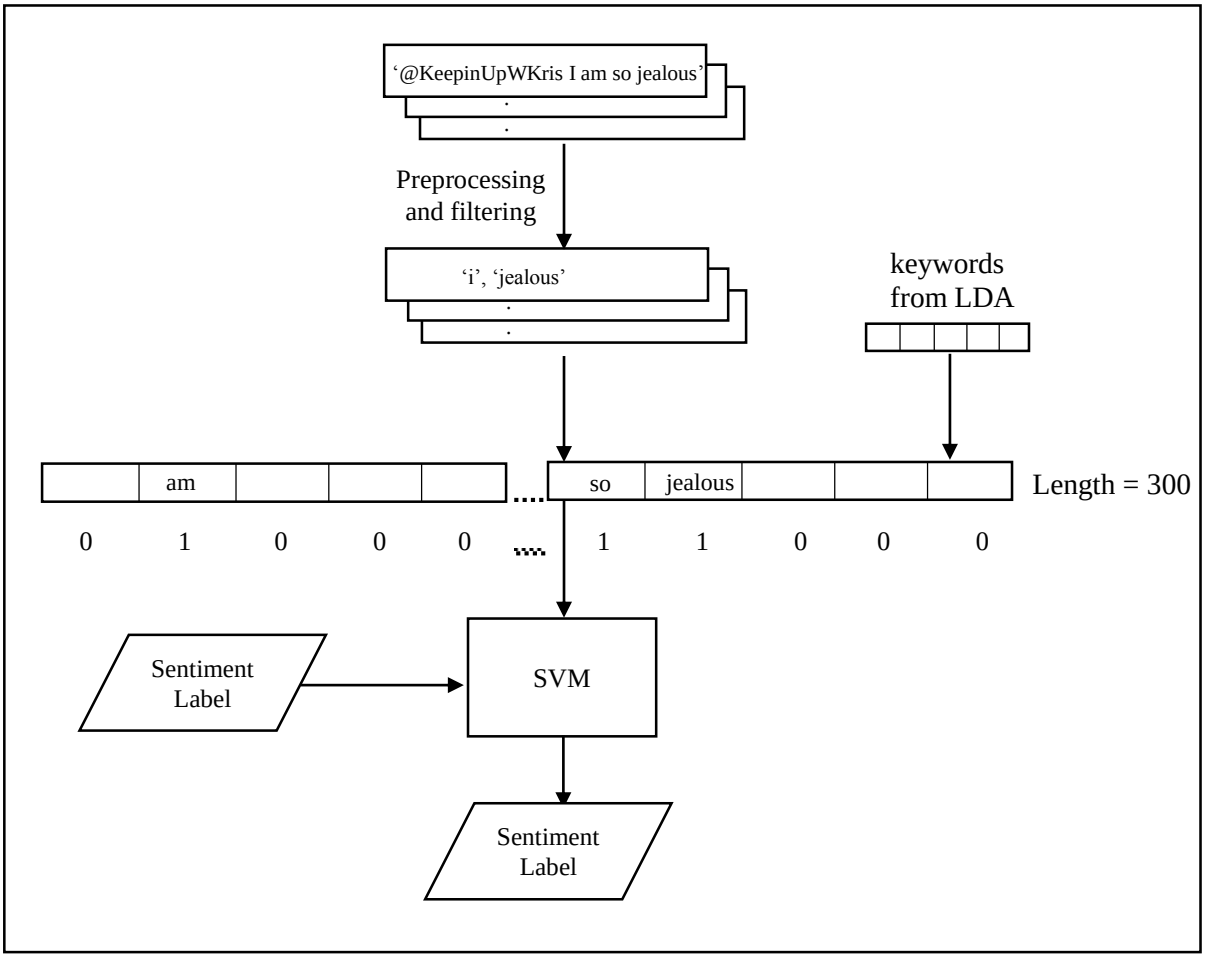

Figure 3.5 Stages of Classification

For classification of tweets, SVM is chosen as the method as it has yielded the best accuracy in classification of movie reviews (Pang, et al., 2002). SVM has less overfitting and is more robust to noise. As for classification, the tool that will be used is LIBSVM (Chang & Lin, 2011). This is because it provides many extensions and necessary classes and functions needed for the classifier. Other than that, it is also an easy script that is easy to understand and implement.

The flow of SVM classification is shown in Figure 3.5. The keywords extracted from the LDA model form the feature list vector of length 300 or 500, depending on the number of keywords. After the training tweets have been preprocessed and filtered, the tweets are then processed again to check if the features of the tweet exist inside the feature list, if it exists, the column value of the unigram features will be set to 1, otherwise 0. With every tweets processed, the value of the feature vector is collected to load into the SVM model together with the sentiment labels of each tweets. The SVM classifier will adjust and identify the optimum hyperplane that differentiates the support vectors as shown in Figure 3.6a and Figure 3.6b.

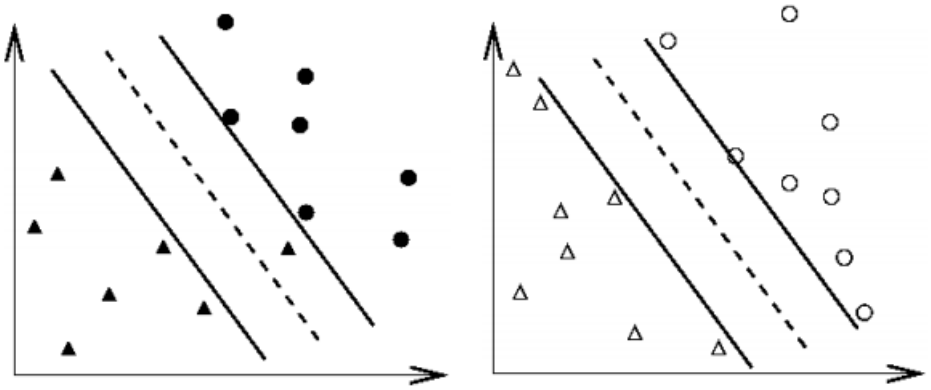

Figure 3.6a Figure 3.6b

Graph showing the support vectors and optimum hyperplane

(▲ and ● are training data, Δ and ○ are testing data)

After training, the model is saved so that it could be used for future classification. During the testing phase, the same process as of the training phase is repeated where the preprocessed and filtered tweets are processed again to get the feature vectors to parse into the trained SVM model. The SVM model then predicts the sentiment of the tweet.

CHAPTER 4: RESULTS AND DISCUSSION

4-1 Experimental Setup

4-1-1 Data Sets

In this project, rather than personally collecting all the tweets from Twitter, dataset is used. Since these datasets are already labelled with their sentiments, it is very convenient to use. These datasets are also commonly used by other researches for their projects and because of that, results obtained by other researches can be used for apple-to-apple comparison during evaluation.

1-1 Stanford Twitter Sentiment Corpus (STS)

This dataset originally contains 1.6 million tweets classified as positive or negative and have been preprocessed to strip off emoticons. The training set consists of 498 tweets that are either positive, negative or neutral. However, in this project, only positive and negative tweets are focused so the neutral tweets will be excluded. Therefore, 60,000 tweets are randomly selected from the original dataset whereby half of the tweets are positive and the other half are negative. The test set on the other hand, after excluding the neutral tweets will be 264 tweets in which, tweets from the original dataset will be randomly picked to be in the training set. Such amount of tweets chosen is so that there is consistency when having comparison with other papers. The dataset file format has 6 fields: the polarity of the tweet, the id of the tweet, the date of the tweet, the query, the user of the tweet and the content of the tweet. Example of the tweets is shown in Table 3.2.

Polarity	ID	Date	Query	User	Text
0 (negative)	1467811592	Mon Apr 06 22:20:03 PDT 2009	NO_QUERY	My birch	Need a hug
0 (negative)	1676451572	Fri May 01 22:05:06 PDT 2009	NO_QUERY	yamababy	I hate washing clothes in the laundry room a freakn minutes like a hour...
4 (positive)	1688856647	Sun May 03 12:04:37 PDT 2009	NO_QUERY	jac0bunderme	what a beautiful day in the neighborhood sunshine!!
4 (positive)	1688857193	Sun May 03 12:04:41 PDT 2009	NO_QUERY	LeeyahWay	I love talking to you. Even if its simple words.

Table 3.2: Example of positive and negative tweets in dataset

4-1-2 Preprocessing

One of the main issues when dealing with raw Twitter data is its casualty and informal language expressed by the users. Pre-processing is an essential step to reduce noise of data into a more formal, readable sentence. There are few steps in preprocessing:

- Firstly, converting the tweets to lower case.
- Replace all the URL links in the tweets to 'URL' to make it easier to remove later on.
- Replace all the "@username" in the tweet to AT_USER.
- Replacing all the hashtags with the same word without the hashtag as hashtag may contain useful information.
- Lastly, whitespaces at the start and ending of the tweet or multiple whitespaces are removed.

As an example, the steps in preprocessing the tweet is shown:

Diary on public option falling down diary list at Dkos http://tinyurl.com/km8cpl @dailykos #publicplan #healthreform #singlepayer #p2

Step	Tweet	Method
1	diary on public option falling down diary list at dkos http://tinyurl.com/km8cpl @dailykos #publicplan #healthreform #singlepayer #p2	Convert to lowercase
2	diary on public option falling down diary list at dkos URL @dailykos #publicplan #healthreform #singlepayer #p2	Replace URL links with URL
3	diary on public option falling down diary list at dkos URL AT_USER #publicplan #healthreform #singlepayer #p2	Replace @user with AT_USER
4	diary on public option falling down diary list at dkos URL AT_USER publicplan healthreform singlepayer p2	Replace hashtag with exact words
5	diary on public option falling down diary list at dkos URL AT_USER publicplan healthreform singlepayer p2	Remove extra whitespaces

4-1-3 Filtering

After preprocessing, the tweets still contain a lot of words that do not indicate any sentiment. Therefore, tweets will go through filtering to remove all those words. The tweets are rid of stop words – a, the, with, is and also ‘AT_USER’ and ‘URL’ from preprocessing. Then, if the tweet contains repeating characters in a word, it looks for 2 or more repetitions of character and replace with the character itself. After that, punctuations are removed. Words that do not start with an alphabet are also removed.

4-2 System Performance

4-2-1 Feature Extraction

Among the training set tweets, the following topic words were some that were extracted by using LDA. As can be seen from the Table 4.1, 'Topic 5' from the positive sentiment seems to be about music whereas 'Topic 1' seems to be about getting sick with stomachache. The rational behind this model is that grouping words under the same topic and bearing similar sentiment could reduce data sparseness in Twitter sentiment classification and improves accuracy (Saif, et al., 2012).

	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
Positive	Day	Twitter	Happy	Thank	Like
	Beautiful	Fun	Go	New	Music
	Awesome	Glad	Birthday	Phone	Sounds
	Sunny	Please	Cool	Tomorrow	Love
	Great	Hear	Now	Like	Good
Negative	Hurts	Sick	Miss	Cold	Damn
	Wrong	Feel	Sorry	Woke	Bored
	Never	Working	Missing	Sad	Need
	Hates	Hot	Homework	Broke	Back
	Tummy	Don't	Sucks	Coffee	Music

Table 4.1: Extracted keywords from LDA

4-2-2 Classification

Table 4.2 reports the accuracy (A) and three sets of precision (P), recall (R), F measure (F) where one is for positive sentiments, one is for negative sentiments and the last for average of two sentiments. There are two sets of features forming from 30 topics and 50 topics in which each set is branched out to another four set where two are features that was extracted by separating positive and negative tweets and another two are features extracted by mixing both positive and negative tweets.

			Positive Sentiments			Negative Sentiments			Average		
	Features	A	P	R	F	P	R	F	P	R	F
T=30	separated with stopwords	61.4	73.06	29.68	42.22	58.56	89.90	70.92	65.81	59.79	56.57
	separated without stopwords	66.9	75.71	44.63	56.16	63.42	86.86	73.31	69.57	65.75	64.74
	mixed with stopwords	63.9	75.68	35.37	48.21	60.49	89.52	72.20	68.09	62.45	60.21
	mixed without stopwords	66.7	75.36	44.42	55.89	63.28	86.67	73.15	69.32	65.55	62.55
T=50	separated with stopwords	64.5	75.21	37.68	50.21	61.10	88.57	72.32	68.16	63.13	61.26
	separated without stopwords	**68.4**	77.13	47.58	58.85	64.73	87.05	74.25	**70.93**	**67.32**	**66.55**
	mixed with stopwords	65.2	76.79	38.32	51.12	61.55	89.33	72.88	69.17	63.83	62
	mixed without stopwords	66.3	76.34	42.11	54.27	62.69	88.00	73.22	69.52	65.11	63.75

Table 4.2: Results of classification

It can be observed from Table 4.2 that classifier trained from unigrams after the removal of stopwords outperforms the result of the classifier that was trained from unigrams with stopwords. As stopwords are too common in each tweets and they held little meaning, it was a common practice to remove them during preprocessing. On the other hand, the keywords obtained from 50 topics performed better than the keywords obtained from 30 topics. However, there is an exception to the keywords extracted by mixing both positive and negative tweets. The keywords from 50 topics outperforms the keywords from 30 topics by a small margin of 0.4%.

The overall result for the recall of positive sentiments was relatively low. It is assumed that a lot of keywords extracted from the negative tweets of the dataset occurs in the positive tweets therefore many positive tweets were wrongly classified as negative. For example, the negative tweet ‘I don’t love you anymore’ – this project is based on unigrams therefore the keywords that may be extracted would be ‘love’ which may also appear in the positive keywords.

Since the classifier that was trained from unigrams, separately extracted from positive tweets and negative tweets and without stopwords yield the highest result of 68.4% accuracy, 70.93% precision, 67.32% recall and 66.55% F measure, another experiment has been conducted to test the factors affecting the performance. The experiment is made up of using three different set of 500 keywords, which are: using 20 topics with its 25 most contributing keywords, using 50 topics with its 10 most contributing keywords and using 50 random topics of 10 keywords out of 100 topics. The result of the experiment can be observed at Table 4.3. From the table, the usage of 50 topics with 10 keywords for each topic as the features still holds the best results although the difference between using 20 topics with 25 keywords from each topics is 1%. Conversely, the result for having 50 random topics out of 100 topics with 10 keywords from each topics was lowest. The reason for this is because when LDA searches for 100 topics, the topics become sparse and when only 50 topics are taken instead of 100, the data becomes sparser.

Features	Accuracy
20 topics with 25 keywords	67.4
50 topics with 10 keywords	68.4
50 random topics with 10 keywords out of 100 topics	63.2

Table 4.3: The accuracy across all three set of data

4-2-3 Speed

The speed of feature extraction with LDA is considerably slow due to the reason that LDA is an algorithm that takes a chunk of documents, process it and keeps iterating until it reaches a certain amount of iteration. So, depending on the size of the documents, if the size is large, then LDA may take a longer time. There is a solution to increase the speed of LDA which is to run distributed LDA on a cluster of computers. The speed for classification is however faster since the trained model can be saved for future uses, the model does not need to be trained again for every classification.

CHAPTER 5: CONCLUSION AND RECOMMENDATION

5-1 Summary

In this project, the proposed method of using LDA as a feature extractor through topic modelling and SVM as classification is developed to test the performance of how the proposed method would perform and to compare the yielded results with some of the proposed methods by other researchers.

The initial concept behind this proposed method was that all the tweets that yield sentiments are more or less related. For example, positive tweets such as 'I am so happy today!' and 'I love this weather'. The words 'happy' and 'love' are words that hold positive sentiments and that bearing all this words into a group would be able to reduce data sparseness and may help to increase the performance of the classifier.

The result, however, was not very satisfying as the highest accuracy reached for all different type of features used is 66.9%. There are many factors which could affect the performance such as preprocessing words, negations and many more. Another result yielded in this project is that training the classifier without stopwords does perform better training with stopwords. This experiments was conducted to observe if stopwords can be used as discriminative features for specific classification task (Saif, et al., 2012).

5-2 Recommendations for Future Works

The overall performance of the model is not very satisfying. As there are still a lot of factors and problems that need to be taken into consideration, there is still a big space for improvements and amendments to improve the performance of the model. The following list are suggestions and recommendations.

Preprocessing and Filtering

As noticed from the data after preprocessing and filtering, there still exists a lot of words with spelling mistakes or typical short form typing, for example: juz, brutha, motha, em, thinkin, u, ur and etc. Therefore, it is recommended to strengthen the process of preprocessing and filtering to remove more of these 'noise' from the tweets so that there could be better accuracy.

N-grams in feature selections

As this project only works around unigrams, it cannot detect negations such as 'not good' will be divided to 'not' and 'good' in unigrams and 'not' may be a negative feature and good is a positive feature. In bigrams, 'not good' will be a negative feature and may be a key factor to determining a tweet's sentiment.

Subjective vs. objective tweets

This project only covers positive vs. negative tweets. By building a classifier for subjective vs. object tweets, it is useful in detecting and filtering tweets with no sentiments therefore, it can be used to classify neutral tweets.

5-3 Conclusion

Natural language processing is not an easy task. In order for a program to work effectively with natural language input, it is required to have the properties of language, knowledge about the things that is mentioned and linguistic structures. Along with the constant changing of languages and slangs, there will always be problems in sentiment analysis. However, with the current advancement of technology, researchers will be able achieve better performance soon.

REFERENCE

Berwick, R., n.d. [Online]
Available at: http://www.svms.org/tutorials/Berwick2003.pdf

Bird, S., Klein, E. & Loper, E., 2009. *Natural Language Processing with Python.* Sebastopol: O'Reilly Media.

Blei, D. M., Ng, A. Y. & Jordan, M. I., 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research 3,* pp. 993-1022.

Chang, C.-C. & Lin, C.-J., 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology,* 2(3), pp. 27:1-27:27.

Domingos, P. & Pazzani, M., 1997. On the Optimality of the Simple Bayesian Classifier under Zero-One Loss. *Machine Learning,* pp. 103-130.

Go, A., Bhayani, R. & Huang, L., 2009. Twitter Sentiment Classification using Distant Supervision. *Processing,* pp. 1-6.

Hsu, C.-W., Chang, C.-C. & Lin, C.-J., 2003. *A Practical Guide to Support Vector Classification,* s.l.: Department of Computer Science, National Taiwan University.

Kouloumpis, E., Wilson, T. & Moore, J., 2011. *Twitter Sentiment Analysis: The Good the Bad and the OMG!.* s.l., The AAAI Press.

Pang, B., Lee, L. & Vaithyanathan, S., 2002. *Thumbs Up? Sentiment Classification Using Machine Learning Technique.* Philadelphia, Association for Computational Linguistics, pp. 79-86.

Ravikiranj, n.d. *how to build a twitter sentiment analyzer? | ravikiranj.* [Online]
Available at: http://ravikiranj.net/drupal/201205/code/machine-learning/how-build-twitter-sentiment-analyzer

Řehůřek, R., n.d. *gensim: Tutorials.* [Online]
Available at: http://radimrehurek.com/gensim/tutorial.html

Řehůřek, R. & Sojka, P., 2010. *Software Framework for Topic Modelling with Large Corpora.* Valletta, Malta, ELRA, pp. 45-50.

Russell, M. A., 2011. *Mining the Social Web.* Sebastopol: O'Reilly Media.
Saif, H., He, Y. & Alani, H., 2012. *Alleviating Data Sparsity for Twitter Sentiment Analysis.* s.l., s.n., pp. 2-9.

Saif, H., He, Y. & Alani, H., 2012. *Semantic Sentiment Analysis of Twitter.* Boston, MA, USA, Springer-Verlag Berlin, Heidelberg ©2012, pp. 508-524.

Sanders, N., n.d. *Sanders Analytics - Twitter Sentiment Corpus.* [Online]
Available at: http://www.sananalytics.com/lab/twitter-sentiment/

Sentiment140, n.d. *For Academics - Sentiment 140 - A Twitter Sentiment Analysis Tool.* [Online]
Available at: http://help.sentiment140.com/for-students

Speriosu, M., Sudan, N., Upadhyay, S. & Baldridge, J., 2011. *Twitter polarity classification with label propagation over lexical links and the follower graph.* Stroudsburg, PA, USA, Association for Computational Linguistics, pp. 53-63.

StreamHacker, n.d. *Text Classification for Sentiment Analysis - Precision and Recall | StreamHacker.* [Online]
Available at: http://streamhacker.com/2010/05/17/text-classification-sentiment-analysis-precision-recall/

Taboada, M. et al., 2011. Lexicon-Based Methods for Sentiment Analysis. *Computational Linguistics,* pp. 267-307.

Taşcı, Ş. & Güngör, T., 2009. *LDA-based Keyword Selection in Text Categorization.* s.l., s.n., pp. 230-235.

Teh, Y. W., Jordan, M. I., Beal, M. J. & Blei, D. M., 2006. Hierarchical Dirichlet processes. *Journal of the American Statistical Association, Vol. 101, No. 476,* pp. 1566-1581.

Wallach, H. M., 2006. *Topic Modeling: Beyond Bag-of-Words.* New York, ACM, pp. 977-984.

Zhang, Z., Phan, X.-H. & Horiguchi, S., 2008. *An Efficient Feature Selection Using Hidden Topic in Text Categorization.* Washington, DC, USA, IEEE Computer Society , pp. 1223-1228.

APPENDIX

Project Planning

Phase 1: Literature Review

During this phase, journals, research papers and survey papers that is related to opinion mining and sentiment analysis will be analyzed in order to gain more knowledge towards this field.

Time duration: June 2013 to July 2013

Phase 2: Research Methodology

Throughout this phase, methods and algorithms are studied, compared and determined.

Time duration: July 2013 to October 2013

Phase 3: Development and Testing

After determining the methodologies and development tools, the methods will be implemented, tested and evaluated.

Time duration: September 2013 to April 2014

Phase 4: Adjustment and Documentation

Lastly, the results obtained will be collected and compiled for documentation.

Time duration: March 2014 to April 2014

Milestone

	Year/Month	2013							2014			
Phase	**Activities**	**June**	**July**	**Aug**	**Sept**	**Oct**	**Nov**	**Dec**	**Jan**	**Feb**	**Mar**	**April**
1	Search for related papers											
	Research about opinion mining											
2	Study and evaluate algorithm											
	Collect and formulate idea for selected algorithm											
3	Develop algorithm											
	Test and evaluate the accuracy of sentiment analysis											
	Testing and Delivery											
4	Documentation and final report compilation											
	Final adjustment											

To be completed

Completed