University of Windsor
Scholarship at UWindsor

Electronic Theses and Dissertations

2016

ASPECT-BASED OPINION MINING OF PRODUCT REVIEWS IN MICROBLOGS USING MOST RELEVANT FREQUENT CLUSTERS OF TERMS

Chukwuma Ejieh
University of Windsor

Follow this and additional works at: https://scholar.uwindsor.ca/etd

Recommended Citation
Ejieh, Chukwuma, "ASPECT-BASED OPINION MINING OF PRODUCT REVIEWS IN MICROBLOGS USING MOST RELEVANT FREQUENT CLUSTERS OF TERMS" (2016). *Electronic Theses and Dissertations*. 5728.
https://scholar.uwindsor.ca/etd/5728

This online database contains the full-text of PhD dissertations and Masters' theses of University of Windsor students from 1954 forward. These documents are made available for personal study and research purposes only, in accordance with the Canadian Copyright Act and the Creative Commons license—CC BY-NC-ND (Attribution, Non-Commercial, No Derivative Works). Under this license, works must always be attributed to the copyright holder (original author), cannot be used for any commercial purposes, and may not be altered. Any other use would require the permission of the copyright holder. Students may inquire about withdrawing their dissertation and/or thesis from this database. For additional inquiries, please contact the repository administrator via email (scholarship@uwindsor.ca) or by telephone at 519-253-3000ext. 3208.

ASPECT-BASED OPINION MINING OF PRODUCT REVIEWS IN MICROBLOGS USING MOST RELEVANT FREQUENT CLUSTERS OF TERMS

By

Chukwuma EJIEH

A Thesis

Submitted to the Faculty of Graduate Studies through the School of Computer Science in Partial Fulfillment of the Requirements for the Degree of Master of Science at the

University of Windsor

Windsor, Ontario, Canada

2016

© 2016 Chukwuma Ejieh

ASPECT-BASED OPINION MINING OF PRODUCT REVIEWS IN MICROBLOGS USING MOST RELEVANT FREQUENT CLUSTERS OF TERMS

By

Chukwuma Ejieh

APPROVED BY:

Eugene H. Kim

Department of Physics

Luis Rueda

School of Computer Science

Christie I. Ezeife, Advisor

School of Computer Science

May 25, 2016

AUTHOR'S DECLARATION OF ORIGINALITY

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.

ABSTRACT

Aspect-based Opinion Mining (ABOM) systems take as input a corpus about a product and aim to mine the aspects (the features or parts) of the product and obtain the opinions of each aspect (how positive or negative the appraisal or emotions towards the aspect is). A few systems like Twitter Aspect Classifier and Twitter Summarization Framework have been proposed to perform ABOM on microblogs. However, the accuracy of these techniques are easily affected by spam posts and buzzwords.

In this thesis we address this problem of removing noisy aspects in ABOM by proposing an algorithm called Microblog Aspect Miner (MAM). MAM classifies the microblog posts into subjective and objective posts, represents the frequent nouns in the subjective posts as vectors, and then clusters them to obtain relevant aspects of the product. MAM achieves a 50% improvement in accuracy in obtaining relevant aspects of products compared to previous systems.

DEDICATION

To my parents, Prof. Mike and Ngozi Ejieh.

ACKNOWLEDGEMENT

My sincere appreciation goes to my parents. Your perseverance and words of encouragement gave me the extra energy to see this work through.

I will be an ingrate without recognising the invaluable tutoring and supervision from Dr. Christie Ezeife. Your constructive criticism and advice at all times gave me the needed drive to successfully complete this work. I like to thank Dr. C. I. Ezeife for the research assistantship positions from Natural Science and Engineering Research Council of Canada (NSERC).

Special thanks go to my external reader, Dr. Eugene H. Kim, my internal reader, Dr. Luis Rueda and the chair, Dr. Richard Frost for accepting to be on my thesis committee. Your decision, despite your tight schedules, to help in reading the thesis and providing valuable input is highly appreciated.

I appreciate the support and encouragement of my family. And lastly to friends and colleagues at University of Windsor, I say a very big thank you for your advice and support throughout the duration of this work.

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

CHAPTER 1: INTRODUCTION

The opinion of a product is what people think about that product. For example, consider the following sentences:

S1: *the iphone is a **great** phone!*

S2: *the iphone is a **horrible** phone!*

S3: *the iphone is a phone*

S1 gives a positive opinion about the iphone, **S2** gives a negative opinion about the iphone and **S3** gives a neutral opinion about the iphone. The positivity, negativity or neutrality of an opinion are all called the ***polarity of the opinion*** (Pang and Lee 2008). From the example above, **S1** and **S2** are called ***subjective sentences*** because they express a positive or negative opinion and **S3** is called an ***objective sentence*** because it expresses a neutral opinion on the iphone. The words '*great'* and '*horrible'* in the first two sentences are referred to as ***opinion words*** because they determine the opinion polarity of the product, *iphone*. Opinion words are mostly adjectives (Liu 2012).

Assuming a producer of a product wants to know what customers think of his/her product to know if they like it or not, the producer may result to creating opinion polls, surveys or forming focus groups. This can be expensive, time intensive and labour intensive. These bring up the need for an automated way of obtaining opinions – Opinion Mining. Opinion mining of products is the field of study that analyzes and discovers people's opinions towards products (Liu 2012). The fast paced growth of web applications such as blogs, forums and review sites have created an easily accessible and fast way for consumers to generate and share opinions about products thus providing massive sources for mining opinions (Pang and Lee 2008). Examples of Opinion Miners are OpinionMiner (Jin et al. 2009) and OPTIMISM (Silva et al. 2009). The basic way these opinion miners work is for each sentence, *S*, they identify the opinion words in each *S* and the opinion polarity of the opinion words determines if the sentence is a positive or negative sentence. For example consider the sentences below:

S4: I like my new car. It is awesome.

S5: I hate my new car. It is horrible

The opinion words in S4 and S5 and their polarity is shown in the table below:

Sentences	**Opinion Words**	**Polarity**
S4	"like", "awesome"	Positive
S5	"hate", "horrible"	Negative

Table 1: Polarity of S4 and S5

The output of the opinion miner will be positive for S4 and negative for S5. To obtain the polarity of opinion words, basically, a tool known as a ***sentiment lexicon*** is used. A sentiment lexicon is a list (or dictionary) that contains words and classifies each word as positive, negative or neutral. An example of a sentiment lexicon is SentiWordNet (Esuli and Sebstaini, 2006). This general form of opinion mining is referred to as ***Document-level Opinion Mining*** (Liu 2012).

Furthermore, assuming the producer wants to know more about his/her product and is not too satisfied just knowing if the customer's like the product or not but wants to know what part of the product they like or hate or what feature of the product they like or hate, the producer needs to perform a form of opinion mining called ***Aspect-based Opinion Mining*.** The parts and features of the product are referred to as the ***aspects*** of the product.

1.1 ASPECT-BASED OPINION MINING

According to (Feldman 2013), Aspect-based Opinion Mining (ABOM) is the research problem that focuses on the recognition of all opinions expressed within a given document about a product and the aspects to which they refer. ABOM systems take as input a body of text about a product and outputs all the aspects or features of the product from the body of text and its respective opinion polarity. A broad overview of the steps in ABOM (these are discussed in greater details in Chapter 2) are:

1. Get a collection of documents about a product that contain customers' opinions. The source of these documents can be product review sites (e.g. amazon.ca and yelp.com) or social media sites on the web.

2. Since aspects of products are majorly nouns (Liu 2007), all the nouns in the collection of documents are obtained.
3. Since there will be a lot of nouns obtained that do not relate to the product, a form of pruning is done to get the relevant aspects of the product.
4. The nearest opinion word to each relevant aspect is obtained and the opinion polarity of the opinion word is obtained. This becomes the opinion on the aspect of the product.

For example, if these two sentences form a collection of documents about a product (iphone):

S1: I love the battery of my iphone. The camera is also good

S2: I hate the screen of my iphone. The charger is also awful

The underlined words are the nouns in each of the sentences (excluding the product name). The table below shows the nouns and the nearest opinion word to each noun:

Sentence	**Aspect**	**Opinion Word**	**Polarity**
S1	battery	Love	positive
	camera	good	positive
S2	screen	hate	negative
	charger	awful	negative

Table 2: Opinion Words in S1 and S2

The output of performing ABOM on S1 and S2 go thus:

Product: *iPhone*	
Aspects	***Opinion polarity***
battery	positive
camera	positive
screen	negative
charger	negative

Table 3: Output of performin ABOM on S1 and S2

Examples of ABOM systems are Red Opal (Scaffidi et al. 2007) and Opinion Digger (Moghaddam and Ester 2010). The major steps taken in ABOM are shown below:

1. **Post Collection -** Obtain a collection of posts about a product. The source of the posts can be Product Review sites, News Articles, Blogs and any other online forums where people talk about products. In this research, Microblogs (Twitter) serves as the source.
2. **Candidate Aspect Collection –** This is majorly done using association rule mining to get the frequently occurring words. The frequently occurring words within the collection of posts are more likely to be an aspect of a product (Liu 2012) hence they are called the candidate aspects.
3. **Candidate Aspect Pruning –** It is at this stage that the "true" aspects of the product are obtained. Existing systems use techniques like Compactness Pruning, TF-IDF and PMI (all discussed in Chapter 2) narrow down the candidate aspects to relevant aspects that are parts or features of the product. This is step is where this study focuses on.
4. **Aspect Opinion Detection –** the opinion polarity of the discovered aspects is obtained at this stage.

The figure below outlines the basic steps taken in performing ABOM in products.

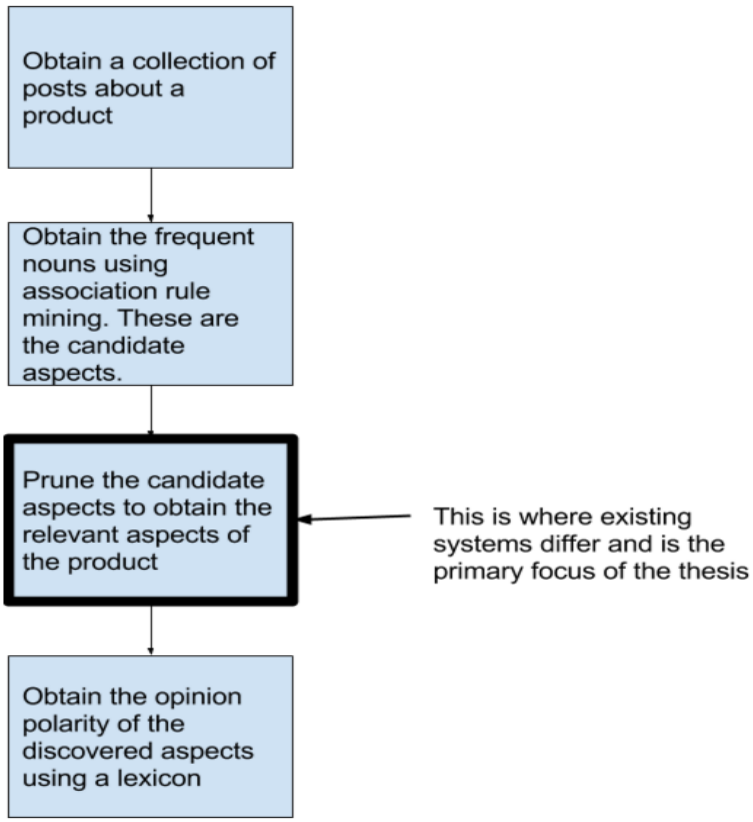


Figure 1: Overview of steps taken in ABOM

1.2 Data Mining

Data mining as defined by (Han, Kamber, and Pei 2012) is "the process of discovering interesting patterns and knowledge from large amounts of data". The data sources from which these patterns can be discovered from can be flat files, relational and transactional databases, data warehouses and the Web. Data mining is used for finding interesting trends or patterns in large datasets to guide decisions about future activities. Data mining is used for information discovery and prediction (Han, Kamber, and Pei 2012). Common data mining approaches are Association Rule Mining, Classification and Clustering.

1.2.1 Data Mining Approaches

The common data mining approaches are:

1) ***Classification***

   In this task, the aim is to predict the label or class for a given unlabeled sample (Zaki and Meira 2014). Formally, given a set of training data with records x1,x2…xN, D = {x1, x2,...,xN} with each record belonging having a label with drawn from a discrete set of class labels, L = {L1, L2,...,Lk}. The task is to construct a classification model such that given an unlabelled test instance, y, the label of y can be got based on features of instances in D and labels in L. For example consider the sample data in Table 1 below that shows information on students and their financial status. The attributes income and credit_rating are called independent attributes and the buy_car attribute is a dependent attribute because it can take only two values "yes" or "no", based on the other two attributes (credit_rating and income).

SN	income	credit_rating	buy_car
1	High	Fair	Yes
2	Medium	Good	Yes
3	Medium	Fair	No
4	High	Bad	No
5	Low	Bad	No
6	High	Good	Yes

Table 4: Sample Data for Classification

The goal of any classification algorithm is to take a training dataset as input and produce a classification model which are rules based on the independent attribute. For example, from the table above, one of the rules for classification is that credit_rating must be either "Fair" or "Good" for buy_car to be "Yes". The classification model learns these rules and uses it to classify a new record of a student which the dependent variable (buy_car) is not known. Some common classification algorithms include k-nearest neighbours (K-NN) which uses a distance measure such as Euclidean Distance to classify new records (Cover and Hart 1967), Naïve Bayes classifier which is a probability based (McCallum and Nigam, others 1998) and Support Vector Machine (SVM) (Cortes and Vapnik 1995). The task of classification has been applied to text mining in the domain of News filtering and Organization, Opinion Mining and Document Organization, Email Classification and Spam Filtering (Aggarwal and Zhai 2012).

2) Clustering

This the process of partitioning a set of data objects into subsets (Han, Kamber, and Pei 2012). This is done in such a way that data objects within a subset or group are very similar and data objects from different subsets or groups are as dissimilar as possible. The similarity of data objects depends on the similarity function used. For example, consider the figure below. The number of words in a document were plotted against the number of

pictures in the document. Each point signifies a document, therefore there are 15 documents in the space. These 15 documents can be grouped into 3 clusters based on how close the points are to each other.

Figure 2: Clustering 15 Documents based on 2 Features

This is an example of the partitioning based clustering paradigm and k-Means algorithm can be used for this. The steps taken to perform k-Means go thus:

i. Pick the number of clusters (k) and choose random k points as centers
ii. Assign each point to the closest center to form partitions
iii. When all objects have been assigned to the nearest center, recalculate the center of each partition
iv. Repeat steps ii and iii until the center does not move again.

The algorithm aims at minimizing the objective function, E, which is:

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} dist(\boldsymbol{p}, \boldsymbol{c_i})^2,$$

Where $\boldsymbol{p}$ is a point and $\boldsymbol{c_i}$ is a centre. So basically, k-Means aims at reducing the sum of the distance between each point and the centre.

Another clustering paradigm is the hierarchical-based clustering. Hierarchical-based clustering can either be *Agglomerative* or *Divisive.* In Agglomerative clustering, each object is taken as its own cluster and then merges with another single cluster. This in turn merges with another related cluster to make a bigger cluster based on a distance metric. This can be seen as a bottom-top approach to clustering. On the other hand, Divisive clustering takes a top-bottom approach in which all objects are placed in a cluster and the cluster is broken down into smaller clusters. Other clustering paradigms are density-based, graph-based and spectral clustering (Zaki and Meira 2014).

3) Association Rule Mining

Association rule mining involves obtaining rules from a given set of items to discover the simultaneous occurrences of different items. It has been used extensively in data mining research where transactions are stored in a structured database and rules of the form x -> y are formed where x, y are items in the database and x is not equal to y. For example, consider the Table 2 below showing a sample transaction history of customers in a grocery store.

Transaction ID	**Purchased Items**
1	Milk, Bread, Butter
2	Milk, Bread
3	Milk, Butter
4	Butter, Bread, Egg, Tea

Table 5: Sample Transaction Data of a Grocery Store

A rule that goes Milk -> Bread means customers who bought Milk also bought Bread. To discover these rules, some algorithms have been proposed such as the Apriori algorithm (Agrawal and Srikant, others 1994) and Frequent Pattern algorithm (Han et. al 2000). The sets {Milk, Bread}, {Milk, Bread, Butter} are all called ***itemsets*.**

Using Apriori algorithm to mine association rules in the table above with a minimum support of 50%, the 1-itemset is first found. This consists of the items *MILK, BREAD,*

EGG, TEA, and BUTTER. Scanning the table above, *MILK, BREAD and BUTTER* appear in two or more transactions. Since the minimum support is 50% and the number of transactions are 4, their support count meets the minimum support so they form the first large itemset, L1. Therefore, L1 = {Milk, Bread, Butter}. Next, the 2-itemset is generated by using the apriori-gen join operator. The apriori-gen join of Li with Li joins every itemset k of first Li with every itemset n of second Li where n > k and first (i-1) members of itemsets k and n are the same. Using this example, applying the apriori-gen join to L1 will give us {Milk-Bread, Milk-Butter, Bread-Butter}. This is the 2-itemset. Since the 3 items meet the minimum support of 50%, they form the second large itemset, L2. Applying the apriori-gen join again to L2 gives {milk-bread-butter} which is our 3-itemset. Since the minimum support for milk-bread-butter is less than the minimum support, the algorithm terminates.

1.2.2 Text Mining

The vast amount of information available to us due to the rise of the World Wide Web has caused a shift of focus from mining and extracting relevant information from structured data sources like relational and transactional databases to mining information from semi-structured and unstructured data sources such as online news feeds, social media, medical reports, email messages and review sites (Grobelnik, Mladenic, and Milic-Frayling 2000). Text mining is considered a natural extension of Data Mining and it aims to give an insight and discover interesting patterns in semi-structured and unstructured data (Sirmakessis 2004). The key element in text mining is the document collection (Feldman and Sanger 2007). A document collection is any group of text-based documents such as news reports, posts on social media and reviews and can be referred to as a corpus (Feldman and Sanger 2007; Manning, Raghavan, and Schütze 2008). The corpus serves as the input in a text mining system.

Another key element in text mining are how the words are represented. The most often used form of representation of words is the ***standard representation of words*** where words are represented as vectors, the length of the vectors is the number of documents in the corpus and the values of the vectors correspond to the frequency of occurrence of each word in a document. For example, if there are 5 documents in the corpus and the word, *dog* appears in only the second document, dog

is represented as [0 1 0 0 0]. A more recent form of representation of words is the ***distributed word representation*** or ***word embeddings***. In this, the words are represented by vectors of a fixed length (usually 100-500) in such a way that words with similar meanings have similar vector representations. The meaning of word is defined by the context of the word (the words that are around the word – the neighbours of a word). So a word like *dog* will be represented with a vector [0.02, 0.04, -0.56 …, 0.84]. The values of the vector are got by using word embedding algorithms like word2vec (Mikolov et al. 2013) which work on a huge corpus of training text.

1.3 Mining the Social Web

(Kaplan and Haenlein 2010) defined social media as "a group of Internet-based applications that build on the ideological and technological foundations of the Web, and that allow the creation and exchange of User Generated Content". According to (Barbier et al. 2013) "it is a conglomerate of different types of social media sites, including social networking (e.g., Facebook, LinkedIn, etc.), blogging (e.g., Huffington Post, Business Insider, Engadget, etc.), micro-blogging (e.g., Twitter, Tumblr, Plurk, etc.), wikis (e.g., Wikipedia, Wikitravel, Wikihow, etc.), social news (e.g., Digg, Slashdot, Reddit, etc.), social bookmarking (e.g., Delicious, StumbleUpon, etc.), media sharing (e.g., Youtube, Flickr, UstreamTV, etc.), opinion, reviews and ratings (e.g., Epinions, Yelp, Cnet, etc.), and community Q&A (e.g., Yahoo Answers, WikiAnswers, etc.)". Table 4 below describes the different categories of social media sites and their key characteristics:

Some of the research areas as regards social media include community detection and analysis, opinion mining and sentiment analysis, social recommendation, influence maximization and modelling, information diffusion and provenance and privacy, security and trust (Gundecha and Liu 2012). In this research we focus on opinion mining in microblogs. Unlike community detection and analysis (Mumu and Ezeife 2014), we are not interested in discovering the social networks formed by a specific set of people, unlike influence modelling (Ahmed and Ezeife 2013), we are not interested in the links and "friends" formed in social media sites and how they influence one another and unlike information diffusion and provenance (Barbier et al. 2013), we are not interested in the origin of the user-generated content social media. In this thesis, our focus is on

the posts made on these microblog sites about products and how we can mine aspects of the products to know the opinions expressed on each of the aspects regardless of who posted them. These microblog posts, collected over a period of time will serve as our document collection (corpus).

Type	Characteristics
Online social networking	Online social networks are Web-based services that allow individuals and communities to connect with real-world friends and acquaintances online. Users interact with each other through status updates, comments, media sharing, messages, etc. (e.g., Facebook, Myspace, LinkedIn).
Blogging	A blog is a journal-like website for users, aka bloggers, to contribute textual and multimedia content, arranged in reverse chronological order. Blogs are generally maintained by an individual or by a community (e.g., Huffington Post, Business Insider, Engadget).
Microblogging	Microblogs can be considered same a blogs but with limited content (e.g., Twitter, Tumblr, Plurk).
Wikis	A wiki is a collaborative editing environment that allow multiple users to develop Web pages (e.g., Wikipedia, Wikitravel, Wikihow).
Social news	Social news refers to the sharing and selection of news stories and articles by community of users (e.g., Digg, Slashdot, Reddit).
Social bookmarking	Social bookmarking sites allow users to bookmark Web content for storage, organization, and sharing (e.g., Delicious, StumbleUpon).
Media sharing	Media sharing is an umbrella term that refers to the sharing of variety of media on the Web including video, audio, and photo (e.g., YouTube, Flickr, UstreamTV).
Opinion, reviews, and ratings	The primary function of such sites is to collect and publish user-submitted content in the form of subjective commentary on existing products, services, entertainment, businesses, places, etc. Some of these sites also provide products reviews (e.g., Epinions, Yelp, Cnet).
Answers	These sites provide a platform for users seeking advice, guidance, or knowledge to ask questions. Other users from the community can answer these questions based on previous experiences, personal opinions, or relevent research. Answers are generally judged using ratings and comments (e.g., Yahoo! answers, WikiAnswers).

Table 6: Characteristics of different classes of social media (Gundecha and Liu 2012)

According to (Jansen et al. 2009), microblogs are short comments and "it is precisely the micro part that makes microblogs unique from other electronic word-of-mouth mediums, including full blogs, web pages, and online reviews." Apart from microblog posts being very short and roughly the length of a typical newspaper headline (Milstein, Sarah et al. 2008). Nakov et al. (2013) further describe microblog posts by stating that "the language they use is very informal, with creative spelling and punctuation, misspellings, slang, new words, URLs, and genre-specific terminology and abbreviations". Consequently, the approaches used in other domains like product reviews from amazon.com, epinions.com and movie reviews performs poorly when applied to microblog posts (Ritter, Clark, and Etzioni, others 2011; Niu, Yin, and Kong 2012; MartíNez-CáMara et al. 2014; Das and Kannan 2014) for reasons we will discuss below.

Figure 3: Sample microblog posts about a camera (Source: twitter.com)

Figure 4: A sample online review about a camera (Moghaddam and Ester 2012)

Figure 4 above shows a typical product review about a camera from a review site. We can notice that some aspects have of the camera (on the left part of the picture) such as *battery life* have been explicitly given and the pros and cons of the camera have already been given. We can notice these are not explicitly given in microblog posts by contrasting Figure 4 with Figure 3 above which shows microblog posts about the same camera. We can also notice the following differences which also pose as challenges:

I. ***Short Text*** - Most ABOM systems for product reviews are trained to work on much more longer texts. The longer the text, the more often aspects of the product are mentioned (Liu, 2007). Hence, it is easy to obtain the aspect of the product by getting the most frequently mentioned nouns in the review text. Conversely, in microblogs, it is common to see posts that no nouns are repeated. Thereby making it harder to obtain the aspect of a product from a post.

II. ***Unconventional Writing*** - Most often in product reviews, the texts are written with proper grammar and the right spelling of words but microblogs take an informal style and misspellings are very common. For example consider this microblog post about an iphone,

"I look at some people music vidz and think Damn you was better shooting that on a iPhone". ABOM systems designed for product reviews will scan for the nouns which are "*music and iPhone"*. The systems will automatically assume the aspect being talked about in the post is Music whereas, the post is talking about the superior camera quality of an iPhone.

III. ***Noisy Tex***t - Microblog posts are full of special characters and also URLs that do not contribute in the task of sentiment analysis. Most often, people use these combination of special characters to express their emotions. For example consider the following posts:

P1: *My new "Alibaba"-branded China-made iPhone cord is possibly the coolest thing you'll see this week. Greeeennnnnnn. ðŸ'š http://t.co/vH6LHMok12*

P2: *Does anyone have an extra cable for iphone?????? Would you mind if I borrow one ????? My iphone is almost dead ☹:'(*

In P1, we have a URL and some special characters and P2 has some special characters at the end. Since some of these special characters show emotions like sadness as in the case of P2, a special form of preprocessing is needed so as not to lose these valuable data. Current ABOM systems perform the basic preprocessing tasks as in section 1.2.2 above which cannot preprocess these special characters and URLs that make up most microblog posts.

IV. ***Implicit Aspects*** - Witty, exaggerated and ambiguous posts and idioms tend to be very common in microblogs. These posts do not explicitly state that an aspect of a product is good or bad and leaves the sentiment orientation expressed on the aspect to be inferred. For example, "*Apparently the new iPhone helps you lose weight. You buy it then you can't afford food for a month"*. This post implicitly refers to the price of the product, iPhone.

V. ***Lack of Opinions*** – In reviews, the writer states a subjective opinion about a particular product so by extension, almost every review has a subjective opinion. This is not the case of microblog posts as some of the posts are news headlines, advertisement and regular sharing of information. For example, consider the following microblog posts:

P3: *#Technews How to Set Up Android Wear for iPhone http://t.co/iMVEG4AHCh*

P4: *iPhone 5 For Sale !!! Everything Legit, Factory Unlocked #seriousinquiriesonly*

P5: *I just bought 1m iPhone Charging Cable - 5 Colours! (now Â£2) via @wowcher https://t.co/7MV4z6UEyc*

P3 is a news headline with the corresponding link to read the full story, **P4** is an advertisement to sell a phone and **P5** is someone informing his network of friends that he just got a new cable. **P3**, **P4**, and **P5** do not express any opinion on the iPhone. ABOM systems used for product reviews are developed on the assumption that each document in the corpus has an opinion, but we can see that is not the case for microblog posts. Hence, they are not useful when mining opinion of aspects in microblog posts.

VI. ***Variations in Naming Conventions*** - product reviews are usually got from sites that cater to a particular geographic region for example, most users of amazon.co.uk are from the United Kingdom so reviews of products on the site are written in British English. But microblogs are global and posts of an aspect of a product can be referred to with different names. For example, it is common for Americans to say "Gas Tank" while the British say "Fuel Tank" when describing the same aspect of a car.

As a result of all these differences and also from previous studies (Ritter, Clark, and Etzioni, others 2011; Niu, Yin, and Kong 2012; MartíNez-CáMara et al. 2014; Das and Kannan 2014), it can be seen that ABOM in microblog domain is more challenging than in product reviews and presents a different problem than that in product reviews.

Thesis Motivation - As mentioned earlier, most studies of Aspect-based Opinion Mining (ABOM) has been done using product reviews as the domain. With over 200 Billion tweets per year and 10,000 tweets per second[1], Twitter has proven to be a suitable platform for gathering opinions about products. There are around 316 million active users[2] per month and businesses are taking advantage of that as they can reach a very wide audience with their products and services and also get some feedback. Businesses perform online reputation management (ORM) to monitor their brand, product and services (Spina et al. 2012). An important task in ORM is knowing what aspects of your brand, product or service customers are talking about at a particular time and what their opinions about the aspects. Also, by knowing the opinions of customer's on these various aspects businesses will be able to fine-tune their product awareness or know how and where to improve on in a particular product. Furthermore, they can also gauge customer's satisfaction and complaints

[1] http://www.internetlivestats.com/twitter-statistics/

[2] https://about.twitter.com/company

and even monitor that of their competitors since microblog posts are open to everyone. It is also helpful when advertising a product because through ABOM, companies will know what aspect of their products customers are most satisfied with in a particular period of time and use that aspect to advertise their products. Furthermore, we use microblog posts as the body of text because:

1. Microblogs are much bigger platforms than online product review sites in terms of number of users and this leads to more opinions expressed on a product.
2. Consumers are more frequently turning to social media such as microblogs to conduct independent searches before making purchasing decisions (Volmer and Precourt, 2008).
3. There is a strong correlation (0.70) between opinions about a product on Twitter and consumers' confidence about that product (European Central Bank, 2014)
4. It is more common for users to offer updates on the performance of products during its lifetime on microblogs and in real-time as opposed to product reviews where the user of the product tends to give a one-time review of the product.

1.3.1 Twitter

With 302 active users per month and over 500 million posts per day[3], Twitter is the most popular microblogging website and it is publicly available to everyone. It has an easily accessible application programming interface (API) that can be used for specific tasks such as downloading tweets (posts made on Twitter). It has gained prominence in the research world as almost every research work on microblogs uses Twitter data. Hence, the term "Microblog" is almost synonymous to Twitter (and will be used interchangeably in this thesis). Below are common terms applicable to Twitter:

- Tweet - This is a Twitter post made by a user and it has a limit of 140 characters
- Username - This identifies a user and it is often preceded with a "@" symbol.
- Hashtag - this is used to tag a tweet to show its relevance towards a particular topic and it is represented with a "#" symbol. For example, #WorldCup

[3] https://about.twitter.com/company

- Follower - when a user A follows user B, he/she can see all the tweets made by the user B. User A is called a follower of user B
- Reply - This is a function that enables a user to respond to a tweet made by another user.
- Retweet - This is a function that enables a user to directly reproduce a tweet made by another user he is following

1.4 Thesis Problem and Contributions

Table 1.5 below shows the huge volume of microblog posts that mention some companies' names so we can see that it is impractical to read each of these posts manually to get what aspects they are talking about and the opinions expressed on each aspect. Hence the problem arises: can we build a system to automatically mine all the aspects from these posts and the opinions expressed on each of the aspects?

Company	Apple	Google	Microsoft	Samsung
Number of posts	115,000	60,000	44,000	39,000

Table 7: Conservative estimates of posts that mention the companies[4]

Current systems that attempt to address a similar problem to ABOM in Microblogs are Twitter Aspect Classifier (TAC) (Lek and Poo 2013) and Twitter Summarization Framework (TSF) (Li and Li 2013)[5].

[4] The posts were collected from Twitter on the 9th of September within a 12-hour window.

[5] The authors did not give the systems any name. Those names are created for ease of discussion

Existing Systems	Research Goal	Technique to Obtain Relevant Aspects	Limitations
Entity Identifier (Spina et. al 2011)	To obtain words from Microblogs that are relevant to the product.	Term Frequency-Inverse Document Frequency (TF-IDF). Based on the frequency of occurrence of a word in posts about a product and posts about a different product.	Accuracy easily affected by spam posts.
Twitter Aspect Classifier (Lek and Poo, 2013)	To obtain opinion polarity of microblog posts about a product based on the opinion polarity of the aspects in the post.	Pointwise Mutual information (PMI). Based on the number of google search results of a word and a product to determine relevancy	Google searches vary by geographic locations and by user
Twitter Summarization Framework (Li and Li, 2013).	To extract the aspects of brands and give a summary of opinions expressed.	Topic Tendency Score (TTS). An extension of TF-IDF by also using the frequency of a word occurring in some certain phrases.	Accuracy easily affected by spam posts.

Table 8: Existing Systems That Perform ABOM in Microblogs

Furthermore, the following shortcomings are common to the existing systems:

i. Competitor's products often come up as aspects of a product. Assuming say we want to mine the aspects of an *iPhone* that are talked about on microblog posts, these systems output terms like *Samsung, Galaxy* as an aspect of an *iPhone.* This is so because most posts that mention the iPhone also mention its competitors like the Samsung Galaxy. So just using the frequency based approaches (explained in section 2.1) without exploring the context of the words is going to always going to output competitor's names which are obviously not aspects of a product.

ii. Also these systems do not explore the relationship between an aspect and a product tend to be affected by trendy news or buzz news. For example, still using the same *iPhone* as the entity and mining its aspects. If a popular world figure like the Pope endorses the iPhone or a picture of him is seen using it, this will create a buzz. Hence, there will be many posts that mention *iPhone* and *Pope*. Since these frequency based systems rely heavily on how often a word is mentioned when mining aspects, they automatically assume *Pope* is an aspect of the *iPhone.* We refer to such aspects as ***noisy aspects*** in this thesis.

iii. These systems are hugely affected by spam posts. Since they mine aspects based on frequency of occurrence, spam words in a spam post that is being reproduced by bots are easily and most often wrongly detected as aspects of an entity by these systems. An example of a spam post is shown below:

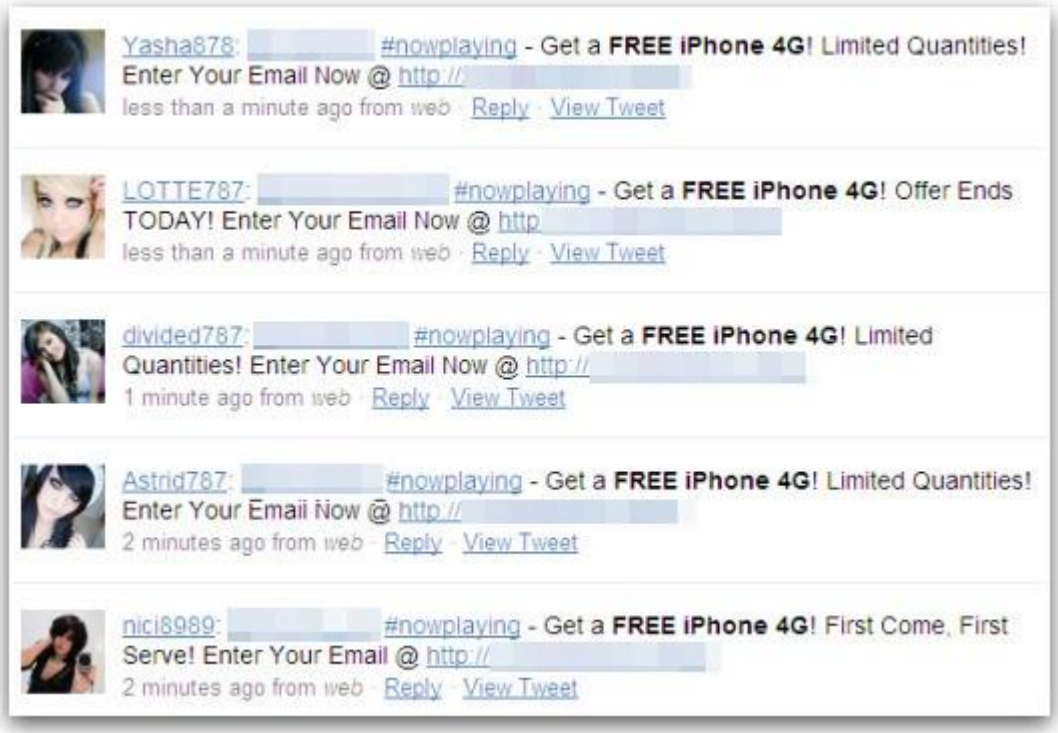


Figure 5: Example of Spam Posts about Iphone

Existing systems that perform ABOM in microblogs use techniques like TF-IDF (Term Frequency – Inverse Document Frequency) and PMI (Pointwise Mutual Information) which are Information Retrieval techniques (Han, Kamber and Pei, 2011) to determine if a word is a relevant aspect of a product. These Information Retrieval techniques deal with ranking the most relevant aspects of product based on a certain score or value. For example, determining if a word is an aspect of a product by using the number of google search results produced by searching the word and the product or by using how less frequently it occurs in posts about other products. On the other hand, this thesis proposes using Data Mining techniques (specifically clustering) to discover interesting patterns and relationships in microblog posts and use these discovered patterns to determine if a word is a relevant aspect of a product or not. We show that clustering of the words will produce more relevant aspects and help deal with the shortcoming of noisy aspects.

Thesis Contributions

A. Thesis Feature Contributions

1. To mine the aspects of a product in such a way that the frequent "noise" in microblog posts does not affect the accuracy of the system. Noise includes:
   i. **Spam** - Posts that are repeated multiple times by a robot
   ii. **Advertisements** - Posts that aim to sell and do not express an opinion on the product. E.g. "*iphone chargers available here for sale. http://asdsahobvvw.com".*
   iii. **Buzz Posts** - Posts are frequently repeated due to a popular event or happening. E.g., "*obama call with an iphone", "obama uses an iphone", "obama caught with iphone".*
   iv. **Competitor's Products** - Posts that mention competitor's products along with the product. E.g., "*samsung upgrades Android on galaxy in light of iPhone release"*
2. To prove that the clustering hypothesis (Rijsbergen 1979) holds in the domain of aspects and products in microblogs. The clustering hypothesis states that "Documents in the same cluster behave similarly with respect to relevance to information needs" (Rijsbergen 1979). Based on this, we form our hypothesis which states that "Relevant aspects of a product belong to the cluster of the product and noisy aspects belong to another cluster". Since clustering involves grouping points in a space based on how ***similar*** (as discussed in section 1.2.1 above) the points

are and we are dealing with microblogs that have a huge volume of stream of words, the following sub-problems arise:

i. The definition of similarity when it comes to products and aspects. For example, how does a computer know that *iphone* is more similar to Apple than to Microsoft?

ii. Converting words into points in a vector space of a fixed dimension and representing the words as vectors.

iii. Representing words as vectors in such a way that semantically similar words have similar vector representations. For example, *iphone* is more semantically similar to *galaxy* than to *BigMac* since they are both phones but the *BigMac* is a food.

B. Thesis Procedure Contributions

To achieve the research goals and to solve the problems, we propose:

1. An algorithm called Microblog Aspect Miner (MAM) which takes in raw and unprocessed microblog posts about a product as the input and outputs the relevant aspects of the products and their opinion polarity. It does this by:
    i. Classifying microblog posts as objective and subjective posts before mining aspects. From observation, it is seen that most of the "noisy" posts do not express a positive or negative opinion on a product (i.e. they are objective posts). For example, "buy your iPhone here" does not give a negative or positive opinion about the iphone. Consequently, getting only the subjective posts helps to eliminate noisy posts.
    ii. Clustering the most frequent nouns using KMeans to remove the noisy candidate aspects and get the true aspects of the product.
2. A Subjectivity Module for calculating the subjectivity score of a post by using the positive and negative scores of words from SentiWordNet (Esuli and Sebastiani 2006). The module takes in a microblog post and determines if the post expresses an opinion on the product or not.
3. A microblog specific model which generates a vector representation for words in microblog posts based on the co-occurrence of words and contexts of words using the word2vec algorithm (Mikolov et al. 2013). The word2vec algorithm is a deep learning

clustering algorithm that is based on matrix factorization and it takes in a collection of documents as input, captures the semantic similarity between words in the collection of documents and outputs the vector representation of each word. We use this algorithm on microblog posts to generate vectors that will serve as the features for our clustering which is done using K-Means.

4. Defining a term called the Aspect-Product Similarity Threshold (APST) which is used to rank how relevant an aspect is to a product. The Aspect-Product Similarity Threshold uses the cosine similarity between the products and the discovered aspects to rank how relevant the aspect is to the product.

1.5 Thesis Outline

In Chapter 2 we provide a detailed related work on ABOM. In Chapter 3 we provide a proposed solution framework with running examples. In Chapter 4 we provide various experimental results including comparisons between the existing and the proposed approach. Finally, Chapter 5 provides some concluding remarks.

CHAPTER 2: RELATED WORKS

In this section, we address related works in text processing and text representations. Furthermore we review systems that address the specific problem of ABOM in microblogs (Li and Li 2013; Lek and Poo 2013; Bahrainian and Dengel 2013; Lim and Buntine 2014) and their shortcomings (section 2.3), systems that address the a related problem of aspect-term extraction or discovery in microblogs which is a major sub-task of ABOM (Zhao et al. 2011; Spina et al. 2012; Das and Kannan 2014), early works on ABOM in general (Hu and Liu 2004b; Hu and Liu 2004a; Popescu and Etzioni 2005; Scaffidi et al. 2007; Moghaddam and Ester 2010) and early works in opinion mining in microblogs (Go, Bhayani, and Huang 2009; Barbosa and Feng 2010; O'Connor et al. 2010; Pak and Paroubek 2010).

2.1 Text Preprocessing

Since most texts are got in an unstructured format, text preprocessing operations help "clean up" the text so they can be fed into the text mining systems. These preprocessing operations include (Miner 2012):

Tokenization

Tokenization is the process of breaking up text-data into tokens. Most often, these tokens are usually words or phrases. For example, the result of tokenizing the sentence, "*I love the new phone that was released by Apple"* is:

[*'I' 'love' 'the' 'new' 'phone' 'that' 'was' 'released' 'by' 'Apple'.*]

Systems used for tokenization are called tokenizers. An example of a tokenizer is Natural Language Toolkit Tokenizer (NLTK, 2015).

Parts-of-Speech Tagging (POS Tagging)

This is the process of assigning parts of speech (e.g. noun, adjective, adverb etc.) to words in a sentence. For example, POS tagging the sentence, "*They refuse to permit us to obtain the refuse permit*" gives the following output:

[*('They', 'PRP'), ('refuse', 'VBP'), ('to', 'TO'), ('permit', 'VB'), ('us', 'PRP'),*

('to', 'TO'), ('obtain', 'VB'), ('the', 'DT'), ('refuse', 'NN'), ('permit', 'NN')]

Where 'NN' is the tag for noun and 'VB' is the tag for verbs. The POS Tags are named according to a naming convention proposed by Santorini (1990) and the complete list of tags and the description is shown in the table below:

POS TAGS	**Description**
CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential there
FW	Foreign word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
NP	Proper noun, singular
NPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PP	Personal pronoun
PP$	Possessive pronoun
RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
SYM	Symbol
TO	to
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present

WDT	Wh-determiner
WP	Wh-pronoun
WP$	Possessive wh-pronoun
WRB	Wh-adverb

Table 9: POS Tags and their Decriptions

An example of a POS Tagger built for microblog posts is Twokenizer (Owoputi et al., 2013).

2.2 Text Representations

To perform a task such as classification or clustering of text documents, the text data is often represented as vectors.

Definition 2.1 - Vocabulary

The ***vocabulary*** of a collection of documents, *V* is all the words contained in the body of text. For example if the following sentences make up the collection, "*I like winter"*, "*Cats like dogs", "I like cats"*, the vocabulary of the collection is:

V = {"I", "like", "winter", "Cats", "like", "dogs"}

The formular for obtaining the vocabulary, *V* of all documents , *D* in a collection, *C* is:

$$V_c = D_1 \mathbf{U} D_2 \mathbf{U} D_{3\ldots} \ldots \mathbf{U} D_n$$

Where *n* is the number of documents in the collection *C*.

Definition 2.2 - Context

The ***context*** of a word is the set of C surrounding words. For instance, the C = 2 context of the word "fox" in the sentence "The quick brown fox jumped over the lazy dog" is {"quick", "brown", "jumped", "over"}.

Definition 2.3 – Semantic Similarity

This is defined using the theory from linguistics which says "Two words that share the same or similar contexts are ***semantically similar"*** (Wittgenstein, 1953). In the posts below, dog is more semantically similar to cat than to ship because they share similar contexts:

P1: the big white ***dog*** sat on the chair

P2: the small white ***cat*** sat on the table

P3: the massive ***ship*** passed under the bridge

From the posts above, we can get the contexts of *dog, cat* and *ship* as:

Context('dog') = ["the", "big", "white", "sat", "on", "the", "chair"]

Context ('cat') = ["the", "small", "white", "sat", "on", "the", "table"]

Context ('ship') = ["the", "massive", "passed", "under", "the", "bridge"]

From above, it is observed that from the above example, the context of *dog* is more similar to the context of *cat* than to the context of *ship*, therefore *dog* is said to be more semantically similar to *cat* than to *ship*.

2.2.1 Standard Representation of Words

In the standard representation of words, words are represented by vectors. The value of the vectors is determined by the frequency of appearance of the word in a document and the length of the vector (dimension of the vector) is the same value as the number of documents in the collection. For example to represent the following steps are taken to represent words in standard representation:

INPUT: A collection, *C* of documents, X_i:

X1: *I like the University*

X2: *Data mining rules*

X3: *I hate the library*

STEP 1

Get the vocabulary, V_c of *C*. This is done by finding the union of all words in the all the documents in *C* i.e. X_1 **U** X_2 **U** X_3. This gives:

V_c = {'I', 'like', 'the', 'University', 'Data', 'mining', 'rules', 'hate', 'library}

STEP 2

For each word in the vocabulary, get its frequency of occurrence in a document. This is shown in the table below. The columns of the table (X_1, X_2, X_3) are the documents in the collection and the rows are each word in the vocabulary and how often they occur in a certain document.

V	X_1	X_2	X_3
i	1	1	0
like	1	0	0
the	1	1	0
university	1	0	0
data	0	1	0
mining	0	1	0
rules	0	1	0
hate	0	0	1
library	0	0	1

Table 10: Vocabulary of the Documents

STEP 3

Each row gives the vector representation of the word. For example, the vector representation of '*the'* is [1 0 0] and that for '*mining'* is [0 1 0].

Some of the shortcomings of the standard representation of words in text mining operations like classification and clustering are:

1. The dimension of the of the vectors increase with the number of documents so it gets computationally expensive to perform operations on it. For example, in the example above, each word is represented by a vector of 3 dimensions in length because there are 3 documents in the collection. If there are 600,000 documents in the collection, each word will be represented as a vector of length 600,000!

2. This representation of words does not capture the semantic similarity between words. For example, it is expected that the vectors that represent "*university*" and "*library*" should be similar because both words are semantically similar (using our knowledge from everyday use of language – natural language). Rather, from the example above it can be seen that the most similar word to '*university'* is '*like'* because both have vector values of [1 0 0].

2.2.2 Distributed Representation of Words

In the distributed representation of words (Bengio et al., 2003), words are represented by vectors of fixed dimensions (usually 100-300) and the values of the vectors are obtained in such a way that words that share similar contexts have similar vector values. Consequently, semantically similar words have similar vector values. To achieve this, the figure below gives a broad overview of the steps taken. A very large collection of words are obtained and they are trained in such a way that words that occur in similar contexts in that collection have similar word vectors (word embeddings).

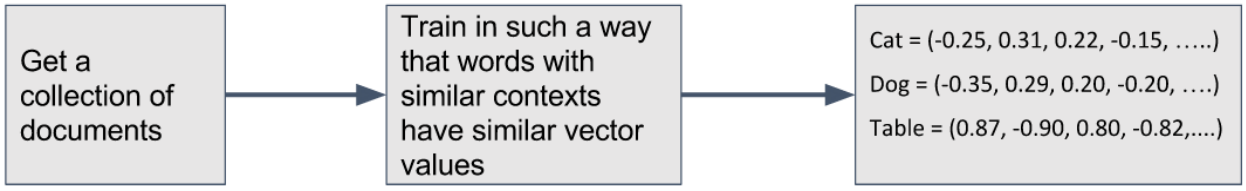


Figure 6: Word Embeddings Overview

An algorithm used for training words in such a way is the word2vec algorithm (Mikolov et al., 2013). The step-by-step details of the algorithm are given below:

Consider the following sentences as making up a collection of words.

S1: *good investment in **stocks** helps the bank generate money*

S2: *money invested in **banks** keep stock prices high*

S3: *the lion ate the **bear** in the zoo*

Take the following steps:

1. Obtain all the words in the collection of documents. This is called the vocabulary.

2. Start with representing each word as a d dimensional vector, d with random small values less than 1. For example, a word, w = [x_1,x_2,... ...x_d] where -1<x<1 and 200<= d <= 500
3. Also represent each context as a d dimensional vector, c just like in step 2 above.
4. Arrange these vectors into 2 matrices, W (which contains all word vectors) and C (which contains all context vectors). Hence the dimension of matrix W becomes |Vw| by |d| and that of matrix C becomes |Vc| by |d|
5. For each sentence segment in the document collection, for each word, get the word vector, w and the context vectors of that word c_1,c_2… …c_{2n} (where n is the number of surrounding words to be used - context window). For example, consider S1 above, if the word we are focusing on is "stocks" (w), the context of that word assuming a context window of 3 are good {good, investment, in, helps, the , bank} which have a context representation of c_1, c_2… …c_6.
6. Modify the values of the vectors in such a way that:
   $\sigma(w.c1) + \sigma(w.c2) + \sigma(w.c3) + \sigma(w.c4) + \sigma(w.c5) + \sigma(w.c6) = 1$
   Where σ is the sigmoid function.
7. Create a corrupt example by choosing a random word (w') from the vocabulary with a different context and using it in the sentence instead of w. For example, a corrupted form of S1 becomes:

   S1': *good investment in **bear** helps the bank generate money*
8. Modify the values of the vectors in such a way that:
   $\sigma(w'.c1) + \sigma(w'.c2) + \sigma(w'.c3) + \sigma(w'.c4) + \sigma(w'.c5) + \sigma(w'.c6) = 0$
9. Repeat steps 5-8 for each row in matrix M.

At the end, it will be seen words that have similar contexts will be represented by vectors of similar values. The goal of this procedure is to obtain vectors to represent each word in the collection such that words that occur in similar contexts (i.e. have similar neighbours) should have similar vector values. The value in each vector is a representation of the value of the word contexts that surround it.

2.3 Existing Systems that Perform ABOM in Microblogs

Twitter Aspect Classifier (TAC)

Lek and Poo (2013) proposed an approach to mine opinions of products from microblog posts. Their system takes in microblog posts (tweets) as input and outputs the corresponding list of possible aspect candidate terms along with their opinions and polarity. To achieve this, the authors make use of a Parts-of-Speech (POS) tagger, a sentiment lexicon (SentiWordNet) and gazetteer lists (a stop word list, a swear word list and an intensifier word list). The POS tagger is used to assign tags (e.g. N to represents Nouns and V to represent Verbs) to each word in the tweet. Each noun is considered as possible aspect term. The authors then use the nearest verb to the left of the noun to determine the sentiment. They check the sentiment lexicon to get the polarity of the verb which in turn becomes the sentiment polarity of the aspect term. For example, consider the following tweet, "*Wind has a very horrible reception today"*. "Reception" which is a noun will be taken as a possible aspect term. "Horrible" is nearest verb to the left of the noun ("reception") so it automatically becomes the sentiment expressed on. The word, 'horrible' is checked in a sentiment lexicon to assign a polarity. In summary, for the above tweet, this step gives an output: *[reception, horrible, -]* which corresponds to the candidate aspect term, the sentiment and the polarity of the sentiment. The gazetteer lists are used in cases of adjectives, adverbs and to correct wrong spellings.

The next step is referred to as Aspect Ranking and Selection where the candidate aspects obtained from the previous stage is ranked to determine the most important aspects. When a collection of tweets regarding a particular entity is retrieved, the number of times a candidate aspect appears is counted. Aspects which fall below a certain threshold that was not stated are ignored. The authors then employ Pointwise Mutual Information (PMI) (Turney 2001) to get the most significant aspects. The PMI value of a pair of aspect-target pair is calculated and the aspect is selected if it has a certain PMI value. The formula for PMI go thus (Turney 2001):

$$\text{PMI}(p,q) = \log_2 \left[\frac{Hits\ (P\ AND\ q)}{Hits\ (p)\ .\ Hits(q)}\right] \qquad (2.1)$$

Where:

p – the product (For example, "*iPhone"*)

q – the aspect of the product (For example, "*Camera*")

Hits(p) - is the number of results that a google query of term "*p"* gives.

For example, if the google search results for "*iphone*" (target product) yields 10 results, the search results for "*camera*" (aspect of target entity) yields 20 results and the search results for "*iphone AND camera*" yields, 5 results, then the PMI value of "*iphone"* and "*camera"* goes thus:

PMI (*iphone, camera)* = $\log_2 (\frac{5}{10 . 20})$.

To explain the system further, we will use a collection of microblog posts about a product (iphone). Consider the following microblog posts below:

P1: *#Technews How to Set Up Android Wear for iPhone http://t.co/JQKQa8PmKE*

P2: *LOT of 140 iPhone 5/C/S Cracked screens with GOOD LCD TESTED! - Full read by eBay: Priceâ€¦*

P3: *Does anyone have an extra cable for iphone?????? Would you mind if I borrow one ????? My iphone is almost deadï¼ • ï¼ • ï¼ • ï¼ • ï¼ •*

P4: *I'm ready for Apple to drop a new iPhone. I'm over this 6 plus ... I want a smaller phone*

P5: *My iphone screen is officially smashed, talk to ya never*

The task is to mine the aspects of the iphone from these collection of microblog posts and determine the opinion polarity on each aspect.

STEP 1

Obtain each noun, abbreviation, @mention or hashtag. These are possible aspects of the product. This gives the list:

SN	Possible Aspect Candidates	Posts in which they occur
1	#Technews	P1
2	Android	P1
3	Wear	P1
4	Screens	P2
5	eBay	P2
6	LCD	P2
7	Cable	P3
8	Apple	P4
9	Phone	P4
10	Screen	P5

STEP II

The nearest adjective, verb or adverb (these will be referred to as modifier) to the left of the aspect candidates obtained in the previous step is chosen. For example, in P1, the nearest modifier (adjective, verb or adverb) to the aspect candidate, "Android" is "set". So set is chosen and the polarity of set (positive or negative) is checked in a lexicon (SentiWordnet). This gives us a results list that contains the possible aspect candidate, the nearest modifier and the polarity of the modifier.

SN	Possible Aspect Candidates	Left-hand Modifier	Polarity of Modifier

1	#Technews	NA	NA
2	Android	set	Neutral
3	Wear	set	Neutral
4	Screens	cracked	Negative
5	eBay	read	Neutral
6	LCD	good	Positive
7	Cable	extra	Neutral
8	Apple	ready	Neutral
9	Phone	smaller	Negative
10	Screen	NA	NA

Table 11: Result List from Step II

STEP III

Similarly, the right side of the possible aspect is also scanned for modifiers as in STEP II above and the polarity of the modifier is also obtained.

STEP IV – Aspect Pruning Stage

The Pointwise Mutual Information (PMI) of each candidate aspect is then calculated for each candidate aspect. The formula to calculate this and how this is done is explained above (pg. 30-31). In the Table 7 below, p refers to the product, q refers to the candidate aspect and "hits" refers to the results got from the google search of the terms.

S/N	q	Hits of q	Hits p AND q	PMI
1	#Technews	185,000	47,500	-32.3545739

2	Android	1,380,000,000	796,000,000	-31.1868759
3	Wear	947,000,000	99,100,000	-33.6494555
4	Screens	594,000,000	39,500,000	-34.3035864
5	ebay	286,000,000	54,100,000	-32.7953627
6	LCD	348,000,000	90,200,000	-32.340936
7	Cable	754,000,000	178,000,000	-32.4757353
8	Apple	1,550,000,000	469,000,000	-32.1176564
9	Phone	4,580,000,000	552,000,000	-33.4456554
10	Screen	1,770,000,000	388,000,000	-32.5826688

Table 12: PMI Score of each Frequent Noun

Assuming we want to get the top 5 aspects of the product, we obtain the highest PMI scores. For this example, the top 5 aspects are *Android, Apple, Screens, LCD,* and *#Technews*. The look up table is checked to get the opinion polarity of these aspects.

Finally, the output of the system will be:

SN	Aspect	Polarity
1	Android	Neutral
2	Apple	Neutral
3	Screens	Negative
4	LCD	Positive
5	#Technews	NA

Table 13: Output of TAC

The shortcomings of this system are:

1. This system uses Google search in the aspect pruning stage to get the relevant aspects of the product and this introduces a form a inaccuracy to the system as search results on google varies according to location, time and user. For example, some search results that come up on google.com will be different than those that come up on google.co.uk. Also, since Google uses user history to modify its search result, the results vary from user to user.
2. Furthermore, using Google search results to calculate the PMI to determine if a candidate aspect is a relevant aspect or not means the system cannot be implemented in a real-time scenario and used by multiple users because Google has a limit to the amount of search requests you can makes per minute.

Twitter Summarization Framework (TSF)

Li and Li (2013) proposes a similar system which mines trendy topics (which can also be aspects) for a brand name or product from microblog posts and obtain the customer's opinions towards that brand name or product. To achieve this, they introduce a trendy topics detection module which discovers the trendy topics or aspects discussed in the collected microblog posts. In this module, for a given query (a query in this case is a topic or aspect related to the product), it assigns a tendency score of how the query is relevant to the product. This is called the Topic Tendency Score (TTS) and it is a variation of the TF-IDF (Term Frequency –Inverse Document Frequency) technique (Salton and McGill 1983). For example, if our product of interest is the iphone and we want to get the opinions of people about the camera (aspect), tweets that mention the word "iphone" are collected and the TTS of the aspect 'camera' is calculated as follows (Li and Li 2013):

$$TTS_{iphone,camera} = TF_{iphone,camera} \times IDF_{iphone,camera} \times MPP_{iphone,camera} \qquad (2.2)$$

Where $TF_{iphone,camera}$ is the number of occurrences of 'camera' in all the tweets about 'iphone' (term frequency), $IDF_{iphone,camera}$ is the logarithmic inverse of the number of times 'camera' is mentioned across tweets abut 'iphone' and $MPP_{iphone,camera}$ is the number of times 'camera' appeared in a meronymic pattern (which is the same as a syntax rule as described in section 1.1). For example, '*battery of iphone is not good'* matches the pattern, ASPECT_of_PRODUCT. The authors claimed to have used more patterns that were not stated in the study.

The following steps were taken by this system:

STEP I

Microblog posts about a range of products are obtained and stored in a content database, **C**. For example, the products can be *starbucks, iphone* and *Mercedes.* For the sake of explanation, let all the posts about *starbucks* be **S**, *iphone,* **I** and *Mercedes,* **M.** Therefore our content database, **C = S ∪ I ∪ M.**

STEP II – Candidate Aspect Retrieval Stage

Assuming the aspects of the iphone are to be obtained, all the frequent nouns $f_1, f_2, \ldots \ldots f_n$ in **I** are obtained.

STEP III – Candidate Aspect Pruning Stage

The Topic Tendency Score (TTS) of each of the frequent nouns are obtained using equation 2.2 above. For example, if the TTS for frequent noun, f_n is to be obtained, the following formula is used:

$$TTS_{iphone,f1} = TF_{I,f1} \times IDF_{C,f1} \times MPP_{I,f1}$$

$TF_{I,f1}$ is the number of times that the frequent noun, f_1 appeared in **I** (which is the collection of posts about the iphone). This is known as the Term Frequency. $IDF_{C,f1}$ is the logarithmic inverse of the number of times f_1 occurred in **C** (which is the collection of posts of all products). This is known as the Inverse Document Frequency. $MPP_{I,f1}$ is the number of times f_1 occurs in **C** with a certain syntactic pattern divided by $TF_{I,f1}$. For example, assuming the frequent noun, f_1 is "screen", using the information below, we can calculate the TTS of screen:

# of posts in content database, **C**	500,000
# of posts about iphone, **I**	200,000
# of posts "screen" appears in **I**	2,000
# of posts "screen" appears in **C**	2,500
# of times "screen" occurs in a certain syntactic pattern in **C**	300

Table 14: Sample Meta Data for Collection of Posts

$$TTS_{iphone,screen} = (200{,}000) \times (\log(\frac{500{,}000}{2{,}500})) \times (\frac{300}{200{,}000})$$

$$= 1589.4952$$

STEP IV – Aspect Ranking Stage

To choose the top 5 aspects of the product, sort the TTS score of all frequent nouns in descending order and the frequent nouns corresponding to the top 5 TTS scores are taken as the aspect of the product.

The shortcomings of this work are:

1. It needs more than one collection of posts about different products to perform TF-IDF which is the major technique it uses. In the example used, it collected posts about 3 products (*iphone, starbucks and Mercedes*). Now, assuming the 3 products are *iphone, nokia and Motorola*, the TF-IDF results will be much different because these 3 products have common aspects (e.g. battery, screen and camera). Consequently, these will have low TTS scores and will not be identified by the system as aspects of the product. Furthermore, this makes it impractical to implement in real-time.
2. It uses syntax patterns (like PRODUCT has an ASPECT) to show that a word is an aspect of a product. For example, the post "*An iphone has a battery"* matches the syntax pattern, PRODUCT has an ASPECT. These syntax patterns are inexhaustive and also the

unconventional forms of writing in microblogs makes these patterns unreliable. Therefore, looking for such patterns in microblog posts is inefficient.

Summarization of Twitter Data

Bahrainian and Dengel (2013) propose an aspect-based summarization system for microblog posts which has an aspect detector module. The following steps are taken in the aspect detector module:

STEP I

Obtain microblog posts about a product.

STEP II

Remove all opinion-bearing words from all the posts. This is done by using a lexicon. Example of opinion-bearing words are *love, like* and *hate.*

STEP III

Remove all valence shifters and stop words using a dictionary. Examples of valence shifters are *but, although, however* etc.

STEP IV

Remove all words from the dictionary except for domain specific words. Choosing domain specific words is done with the help of a manually edited dictionary.

STEP V

Remove all terms that relate to the competitor's products.

STEP VI

Obtain the most frequent words that occur in the collection of posts. The obtained words are the aspects of the product.

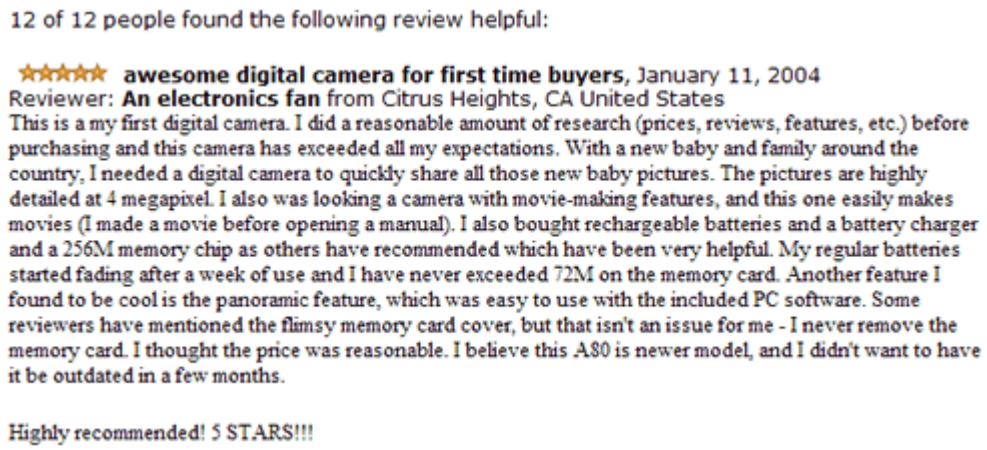
The shortcomings of this study is that most of the tasks are manually done so it will be impractical to scale it to a large number of microblog posts. In this thesis, the tasks will be automated.

2.4 Studies on ABOM in Product Reviews

(Hu and Liu 2004b) were among the first to introduce and address the problem of ABOM. As with all of the early studies on the problem, this was done in the domain of product reviews. The authors took the following steps:

STEP 1

Get a collection of product reviews about a document from a product review site such as amazon.ca. This serves as the input data. An example of a product review is shown below:


12 of 12 people found the following review helpful:

☆☆☆☆☆ **awesome digital camera for first time buyers**, January 11, 2004
Reviewer: **An electronics fan** from Citrus Heights, CA United States
This is a my first digital camera. I did a reasonable amount of research (prices, reviews, features, etc.) before purchasing and this camera has exceeded all my expectations. With a new baby and family around the country, I needed a digital camera to quickly share all those new baby pictures. The pictures are highly detailed at 4 megapixel. I also was looking a camera with movie-making features, and this one easily makes movies (I made a movie before opening a manual). I also bought rechargeable batteries and a battery charger and a 256M memory chip as others have recommended which have been very helpful. My regular batteries started fading after a week of use and I have never exceeded 72M on the memory card. Another feature I found to be cool is the panoramic feature, which was easy to use with the included PC software. Some reviewers have mentioned the flimsy memory card cover, but that isn't an issue for me - I never remove the memory card. I thought the price was reasonable. I believe this A80 is newer model, and I didn't want to have it be outdated in a few months.

Highly recommended! 5 STARS!!!

Figure 7: A Sample Product Review

STEP 2

Each word in each review is assigned a parts of speech tag using POS tagger called NLProcessor (NLProcessor, 2001) to obtain all nouns. For example, assigning POS Tags to the sentence, '*I am amazingly in awe of this my new phone*" gives:

('I', 'PRP'), ('am', 'VBP'), ('amazingly', 'RB'), ('in', 'IN'), ('awe', 'NN'), ('of', 'IN'), ('this', 'DT'), ('my', 'PRP$'), ('new', 'JJ'), ('phone', 'NN')

STEP 3

Apriori algorithm with a minimum support of 1% is applied used to obtain the frequently occurring nouns this are referred to as candidate aspects. For example, assuming there are 4 sentences in the review, the frequently occurring nouns are shown below

Sentence #	Noun/Noun Phrase
1	camera, the focus, manual, a broad strap
2	the memory card, lens,
3	bright pictures, camera, zoom
4	auto-focus, camera, lens,

Table 15: Sample Structure of Transaction File

STEP 4

Two stages of pruning are done to the candidate aspects to get the aspects of the product:

Compactness Pruning - checks candidate aspects that are more 2 words and removes those not compact. For example 'digital camera' is not compact in S3 below but compact in S1 and S2.

S1: "I had searched for a digital camera for 3 months."

S2: "This is the best digital camera on the market"

S3: "The camera does not have a digital zoom"

Redundancy Pruning - removes candidate aspects with a p-support less than 3. For example:

The noun disk, is a subset of phrases like disk drive and disk lens. If the noun disk, appears 10 times in the collection of reviews and out of that 10 times, it occurs as a subset of phrases 7 times, then disk as a p-support of 3.

STEP 5

The nearest adjective (modifier) to the candidate aspect is obtained. For example, consider the sentence, "The **strap** is *horrible* and gets in the way of parts of the camera you need access to". If *strap* is an aspect of the product, the nearest adjective to it is *horrible*.

STEP 6

The polarity (positive or negative) of the modifier is obtained by checking WordNet (Felbaum, 1998). For example, the polarity of *horrible* (the modifier of strap) is negative according to WordNet. So the opinion on *strap* is negative.

Assuming the product reviews are about a camera, after mining the aspects and determining the polarity, the output of the system looks like this:

Aspect	*Opinion*
Lens	Positive
Price	Negative
Zoom	Negative
Battery	Positive

Table 16: Output of the System

Popescu and Etzioni (2005) introduce a system that addresses the problem of ABOM in product reviews called OPINE. Given a particular product and a corresponding set of reviews, OPINE identifies the product aspects, the opinions regarding the product aspects, the polarity of the opinion and ranks the opinions. The figure below gives an overview of what OPINE does:

Input: product class C, reviews R.
Output: set of [feature, ranked opinion list] tuples
R' ← parseReviews(R);
E ← findExplicitFeatures(R', C);
O ← findOpinions(R', E);
CO ← clusterOpinions(O);
I ← findImplicitFeatures(CO, E);
RO ← rankOpinions(CO);
$\{(f, o_i, ...o_j)...\}$←outputTuples(RO, I ∪ E);

Figure 8: Overview of OPINE (Popescu and Etzioni 2005)

To mine the aspects that are explicitly expressed in the review, the authors use the frequent nouns approach introduced by Hu and Liu (2004) (which was reviewed above) but they incorporate Point Mutual Information (PMI) (P. Turney 2001) between phrases that are extracted from a web search engine as explained in section 2.1 above. The authors state that opine achieved a 22% increase in precision of mining aspects of products by using PMI statistics when compared with the approach of Hu and Liu (2004).

Scaffidi et al. (2007) introduce Red Opal which is a system that examines customer reviews, identifies product aspect and scores each product on each feature with the aim of recommending products to users based on the score of each aspect. Red Opal is made up of a Feature (aspect) Extractor module and a Product Scorer module. In the feature extractor module, the review text is examined and the aspect of the product is examined. To do this, the authors extended the approach of Hu and Liu (2004) by working on the assumption that some nouns occur more frequently in review texts than in a generic selection of English text of equal length. Consequently, they use the probability of an identified noun being in a random body of English text to identify aspects.

Moghaddam and Ester (2010) introduce a system called Opinion Digger which mines important aspects of a product and measures the customers' satisfaction of the product on a scale of 1-5. The system takes reviews from epinions.com, a set of predefined aspects and a rating guideline as input and outputs a set of additional aspects and the estimated rating of each product. The figure below describes the input and output of Opinion Digger:

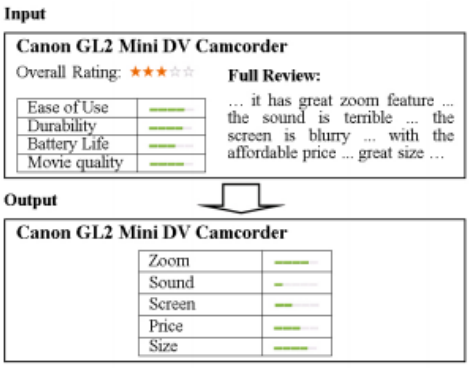


Figure 9: Input and Output of Opinion Digger (Moghaddam and Ester 2010)

Opinion Digger consists of two phases: extracting product aspects and estimating aspect ratings. To extract product aspects, the authors use the approach of Hu and Liu (2012) by finding frequent

nouns and noun phrases. In addition to that, unlike the previous work, the authors also mine opinion patterns and filter out non-aspects. To mine opinion patterns, Opinion Digger searches each known aspect in the review and looks for the nearest adjective to the known aspect. The POS tags of all words between the nearest adjective and the known aspect is taken as a pattern. It then looks for a similar pattern in other reviews to get more aspects of the product. To obtain the estimated aspect ratings, Opinion Digger obtains the sentiment which is the nearest adjective to the discovered aspects, then using k-Nearest Neighbour algorithm (Cover and Hart 1967), it computes the similarity between the discovered adjectives and other words in WordNet. So the rating of the sentiment *snt* is given by (Moghaddam and Ester 2010):

$$\boldsymbol{r_{snt}} = \sum_i \boldsymbol{w_i} \times \boldsymbol{r_i} \div \sum_i \boldsymbol{w_i} \qquad (2.5)$$

Where $\boldsymbol{r_i}$ is the rating of the neighbour and $\boldsymbol{w_i}$ is the inverse of the distance between the sentiment and the neighbour.

2.5 Other Studies in Microblogs

Zhao et al. (2011) is one of the early studies on this sub-task when they addressed the problem of mining topical key phrases for summarizing and analyzing microblog posts. These topical key phrases can also be referred to as aspects. The authors use three steps for key phrase extraction - Keyword Ranking, Candidate Key phrase Generation, Key phrase Ranking. To detect the topics or candidate key phrases, the authors propose using Twitter-LDA. Latent Dirichlet Allocation (LDA) (Blei, Ng, and Jordan 2003) is a method used to find topics in a corpus. Twitter-LDA is a model developed by the Zhao et al. [2011] and it assumes a single topic assignment for an entire tweet. For keyword ranking the authors propose modifying Topical PageRank (TPR) by introducing topic-sensitive score propagation. Topical PageRank runs topic-biased PageRank separately for each topic. They call this method, context-sensitive topical PageRank (cTPR). For key phrase ranking, they propose a principled probabilistic phrase ranking method, which can be flexibly combined with any keyword ranking method and candidate key phrase generation method.

Spina et al. (2012) also addressed this by formulating the problem as an information retrieval task where the goal is to provide a ranking of terms extracted from tweets that are relevant to the company. To achieve this, the tweets relevant to an entity need to be identified and these tweets

then need to be analyzed in order to identify aspects. The authors propose preprocessing the tweets by changing to lowercase, removing punctuations and tokenizing them. They then propose adopting four methods from information retrieval and opinion target identification to address the task of identifying aspects within the tweets. The methods are TF-IDF, log-likelihood ratio (LLR), parsimonious language models (PLM) and an opinion-oriented method (OO) that extracts targets of opinions to generate a topic-specific sentiment lexicon. The authors state that all methods are to follow the same principle: comparing a pseudo document built from entity-specific tweets with a background corpus. They found out that TF-IDF method performs better than the other methods used in their study. They use the following scoring function, s in their study (Spina et. al 2012):

$$s(t, D, C) = tf(t, D) \cdot \log \frac{N}{df(t)} \qquad (2.3)$$

Where *tf* is the term frequency of term *t* in document, *D*; *df(t)* is the number of documents that contain term, *t* and *N* is the total number of documents in the collection, *C.* For example, considering a collection of 20 documents (microblog posts) about the iPhone, if the word "battery" occurs in 15 of those documents only once, the TF-IDF score can be calculated as:

S(battery, Documents, Collection) = 15 × log (20/15) = 4.3152.

Das and Kannan (2014) address the problem of discovering aspects from microblog posts. They propose an algorithm that automatically discovers a ranked list of top aspects of a target entity. The authors propose an approach in which identification of candidate phrases for aspects is the first step. This is done by using a Twitter-specific part-of-speech (POS) tagger to identify a candidate set of noun phrases. A POS tagger is a tool that assigns a part of speech (noun, adjective, verb etc.) to each word in a body of text. For each phrase, the authors propose computing a global indicator - *uniqueness*. This indicator measures how strongly the phrase is correlated with the target entity. Two local indicators are also computed - *burstiness* and *diversity*. The former measures the frequency of usage of the phrase and the latter measures how diversely the phrase is used. The authors then propose training a probabilistic model using Expectation-Maximization algorithm to learn if a particular candidate noun phrase is a relevant aspect based on the three indicators mentioned above. To get data to train the model, the authors stated they used the premise

that a phrase that is related to an entity and also popular on the web is a likely candidate aspect of that entity. Based on this they propose issuing each phrase to be labelled as a query to a web search engine and retrieve the top 50 results. They then propose labelling the phrase as an aspect if at least 10% of the top 50 results have webpage titles that are relevant to the entity. The probabilistic model used is a Gaussian mixture model and it captures the relationship among the diversity, uniqueness and burstiness of a phrase.

Go et al. (2009) were among the first to address the problem of opinion mining in microblogs by basically classifying tweets into positive or negative suing a machine learning approach. The authors propose an approach in which they train different machine learning classifiers (Naive Bayes, Maximum Entropy and Support Vector Machines) using tweets that express both positive and negative sentiments. To get the positive and negative tweets, the authors propose querying the Twitter API for tweets that contain happy emoticons like ":)" and sad emoticons like ":(" respectively. The authors propose using unigrams, bigrams, unigrams and bigrams, and unigrams with part of speech tags as features. An n-gram is a sequence of n items that share the same boarder in a body of text. To reduce the amount of features in their training data, they proposed removing Twitter usernames, URL links and repeated letters from the tweets. The authors state they used Twittatr as the baseline for testing the accuracy of the polarity of the training data. Twittratr is a website that performs sentiment analysis on tweets using a list of positive and negative words. The training data was post-processed by removing emoticons, retweets and repeated tweets. They further state that they used a training data size of 1,600,000 to train the classifiers and the test data was manually collected and it consisted of 177 negative tweets and 182 positive tweets.

Pak and Paroubek (2010) used a similar machine learning approach but they employed the use POS tags to compute the posterior probability in their Naïve-Bayes based model for classifying tweets. They use an approach in which they collect positive and negative tweets, and a corpus of objective texts. This is done by querying the Twitter API for happy and sad emoticons. Objective texts are collected automatically by also querying the Twitter API for tweets from newspapers and magazines. Statistical linguistic analysis is performed on the collected corpus. Word frequencies distribution is checked and TreeTagger (Shmid, 1994) is then used to POS tag all posts in the corpus. A sentiment classification system is then built using the collected corpora. Firstly, filtering is to be performed to remove URL links, then tokenization, removal of stop words and constructing

n-grams to follow respectively. Finally, multinomial Naive Bayes classifier is used to build the sentiment classifier.

Barbosa and Feng (2010) classify tweets into positive, negative and neutral also using a machine learning approach. The authors propose a 2-step sentiment analysis classification method for Twitter, which first classifies messages as subjective and objective, and further distinguishes the subjective tweets as positive or negative. To reduce the labelling effort in creating the classifier, the authors leveraged on three sentiment detection websites (Twendz, TwitterSentiment and TweetFeel) to produce the labels on twitter data. For the first step, the authors proposed mapping each word in a tweet to its part-of-speech (POS) using a POS dictionary. The authors used certain POS tags like adjectives and interjections as indicators that a sentiment has been expressed. For the second step, the polarity (positive or negative) of the sentiments expressed in the tweets got from the first step is determined. To do this the quality of the polarity labels provided by the three websites is analyzed and then some feature selection is done to detect the polarity.

Apart from using machine learning approaches to classify opinions in tweets, lexicon-driven approaches have also been employed. **O'Connor et al. (2010)** employ a lexicon-driven approach to classify public opinion from tweets. They use a subjectivity lexicon, OpinionFinder (Wilson, Wiebe, and Hoffmann 2005) which contains 1,600 positive and 1,200 negative words. To get the opinion polarity of a tweet, they count the positive or negative words in the tweet. If the tweet has more positive words, it is classified as positive. Even with this simplistic approach, they report an accuracy in classification of up to 80% in some data sets.

CHAPTER 3: PROPOSED ASPECT-BASED OPINION MINING SYSTEM FOR MICROBLOGS – MICROBLOG ASPECT MINER (MAM)

As mentioned earlier (section 1.5) the problem addressed is dealing with noisy posts and accurately mining aspects of products using a clustering approach. Our proposed approach, Microblog Aspect Miner (MAM) consists of 4 major steps which will be explained in greater details in next 3 sections. The main algorithm for MAM is shown below:

Algorithm: MAM to mine the aspects, *a* of product, *e* from a stream of Twitter posts, *P* **Input**: Product name, *e* which serves as a search query to the Twitter API **Output**: A ranked list of the relevant aspects of the Product, *e* being talked about on Twitter.
BEGIN 1. MAM collects all Twitter posts, *P* about a product, *e* within a period of time 2. MAM calls the preprocessing module. This removes all noisy posts from *P* and keep only the Subjective Posts, *Sp* (section 3.1, Algorithm 3.2) 3. MAM calls the Aspect Mining Module (algorithm 3.3) to obtain the relevant aspects, *a* of the Product, *P* from the subjective posts, S_p 4. MAM calls the Aspect Opining Mining Module to get the opinion polarity of the obtained relevant aspects. **END**

Algorithm 3.1: Main Algorithm for MAM

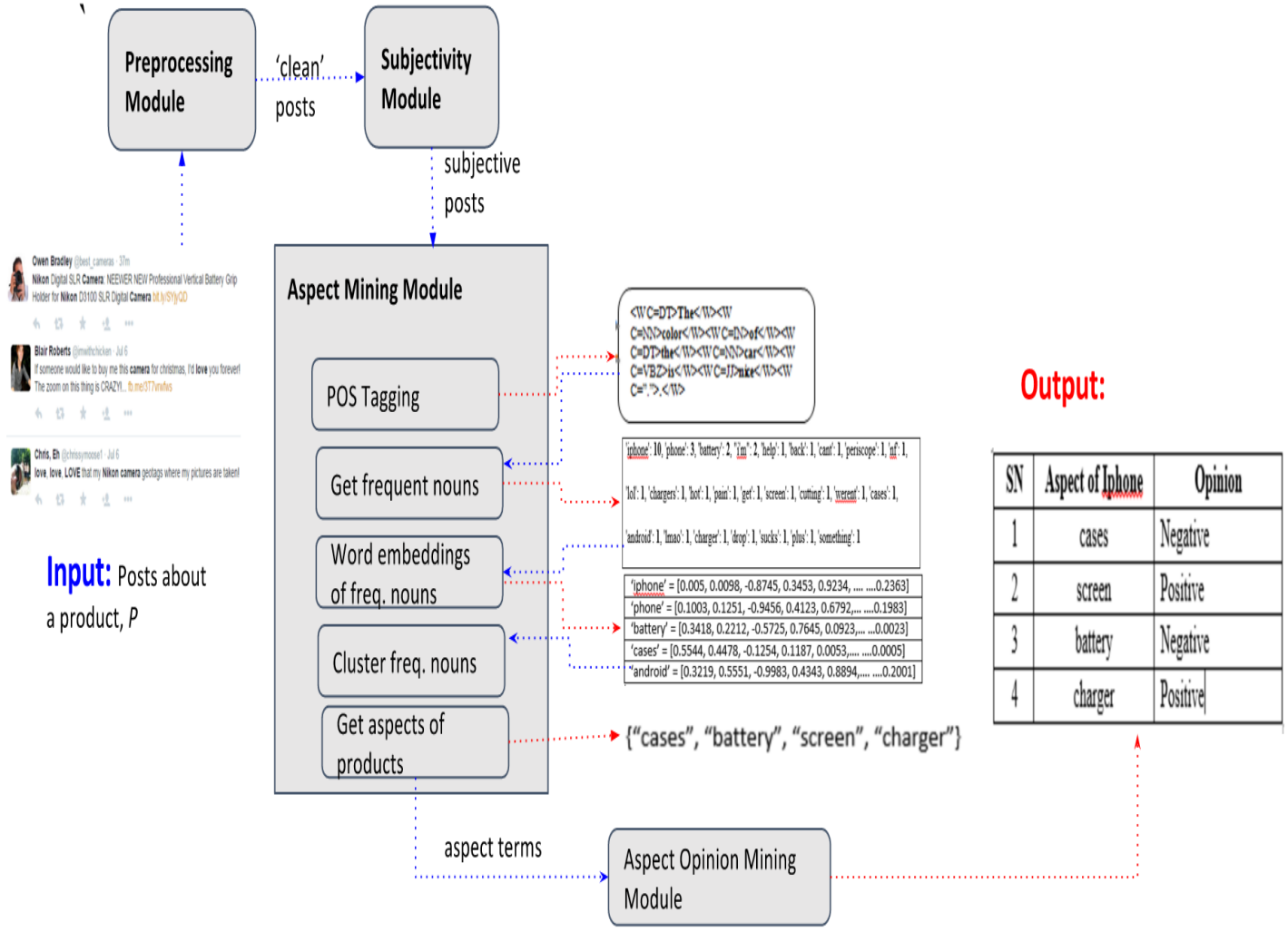


Figure 10: Architecture of the Proposed Aspect-based Opinion Miner

3.1 Removing "noisy" microblog posts

According to Erdmann et al. (2014), 80% of microblog posts about a product are noisy posts. Consequently, it is important that these posts are removed so as to get relevant aspect of the products. To achieve this:

1. We clean up the posts to remove every URL link and foreign characters. This is done in the pre-processing module

2. We introduce a subjectivity module to detect if a post expresses an opinion or not. Posts that do not express are opinion are discarded.

The steps taken in this two modules are discussed in detail in the following 2 subsections.

3.1.1 The Pre-processing Module

This module takes in raw posts about a product and outputs a cleaned version of the post or discards the post based on some rules. The rules are:

i. Any microblog post that had a URL is removed. This is done because it was observed that posts with URLs were advertisements or news headlines, advertisements and spam which will not contribute to the task of mining opinions.

ii. All posts starting with the "RT" letters were also removed as this signifies this is a "retweet". Therefore, that post is being duplicated.

iii. Since we are not concerned about the users and the relationships with the users, we remove every word that begins with the "@" symbol as this is used at the beginning of usernames in twitter. We also strip hash tags ("#") at the beginning of words.

iv. Every word that does not start with a letter of the English alphabet or a digit is removed.

	Original Tweet	Kept for Processing	Processed Tweet
P1	Amazon Prime Video Introduces Offline Viewing for iPhone and iPad http://t.co/BttbFRyfTX Mitchel Broussard	No	-
P2	iPhone 6 are a pain for phone cases ðŸ˜, I mean why make a phone so thin & not bring out good phone cases.	Yes	iPhone 6 are a pain for phone cases, I mean why make a phone so thin; not bring out good phone cases.
P3	RT @SohailThoughts: TOP 5 Reasons to Buy iPhone 6.	No	-
P4	@rowanaelin both bc i have a friend who downloads books using her iphone	Yes	both bc i have a friend who downloads books using her iphone

P5	Lol i like videos and u taking me wth an iPhone,im happy because the pictures are superb SMH #CliveNaidoo vs #MetroCop	Yes	Lol i like videos and u taking me wth an iPhone,im happy because the pictures are superb SMH CliveNaidoo vs MetroCop

Table 17: Preprocessing of Tweets

Table 16 above shows some microblog posts and the corresponding output from the pre-processing module. **P1** and **P3** are discarded because they contain a URL and start with "RT" respectively. As mentioned earlier, "RT" means a retweet and it is often a duplicate of a former tweet. **P2, P4** and **P5** are kept and are "cleaned up" to remove any foreign characters, words that start with a "@" symbol (which signifies a username) and to strip of hast tags ('#').

3.1.2 The Subjectivity Module

According to Liu (2012), subjective statements are those that express a positive or negative opinion while an objective does not express an opinion. Some objective statements can be statements of fact. For example, *"I love the earth"* is a subjective statement but *"The earth is round"* is an objective statement. To accomplish the task of ABOM on Microblogs, we need to obtain subjective posts. The subjectivity module takes in pre-processed tweets and uses opinion scores from SentiWordNet (Baccianella, Esuli, and Sebastiani 2010) to obtain the subjective posts. SentiWordnet is a lexicon used in for opinion mining in which each word is assigned a positive, negative and objective score. Table 17 below shows some words and their corresponding scores in SentiWordNet. It is observed that the sum of the scores for each word sums up to zero. Only the positive and negative scores are used by the subjectivity module to determine if a post is subjective.

Word	Positive Score	Negative Score	Objective Score
Happy	0.875	0.00	0.125
Sad	0.125	0.75	0.125
Bad	0.00	0.625	0.375
Good	0.75	0.00	0.25
Ugly	0.00	0.375	0.625
Beautiful	0.75	0.00	0.25

Table 18: Some words and their scores on SentiWordNet

Algorithm: to obtain subjective microblog posts

Input: Pre-processed posts, *P* from the Preprocessing module

Variables:

pos_score: positive word score from SentiWordNet

neg_score: negative word score from SentiWordNet

Output: Subjective posts, *SP*

START

1 **FOR** post, *p* in pre-processed posts **DO**:

2 Get the *pos_score* and *neg_score* of each word, *w* from SentiWordNet

3 Get the subjective score for *p* (see equation 3.1 below)

4 **IF** the subjective score ≥ 0.04 **DO:**

5 Add post, *p* to collection of subjective posts, *SP*

STOP

Algorithm 3.2: Obtaining Subjective Posts

Calculating the Subjectivity of Posts

To obtain the subjective score in lines 3 and 4 of Algorithm 3.1 above, the following formula is proposed:

$$\text{Subjective Score for post, } p = \frac{\sum_{i=1}^{n} w_i \ (pos_{score} + neg_{score})}{n} \qquad (3.1)$$

Where *w* is a word in *p* and *n* is the number of words in *p*.

Basically, the formula above adds the positive and negative scores for each word in a post, sums this up for each word in the post and divides by the number of words in the post. This was derived on the observation that subjective posts often have some sentiment-bearing words (positive or negative) and often indicate the presence of an opinion. So getting a threshold of the sum of these words will serve as an indicator of subjectivity. Dividing by the number of words in a post serves as a form of normalization. For example consider the two posts below:

P1: *why must my pic take forever to load when I want to use them for snapchat or Twitter? am I the only one with this problem?*

P2: *I love the picture*

It can be clearly seen that **P2** conveys a stronger opinion than **P1** about the picture. But, the summation of the positive and negative score of words in **P1** is higher than **P2** because there are more words in **P1.** This gives a false sense that **P1** conveys a stronger opinion than **P2.** Normalization by dividing by the number of words in the post helps solve this problem. This can be seen in Table 18 below.

	(Word, pos_score, neg_score)	**Summation of scores**	**Subjective Score (Summation divided by number of words)**
P1	('why', 0.0, 0.0) ('must', 0.375, 0.0) ('pic', 0.0, 0.0) ('take', 0.0, 0.0) ('forever', 0.125, 0.125) ('load', 0.0, 0.0) ('I', 0.0, 0.0) ('want', 0.0, 0.25) ('use', 0.0, 0.0) ('or', 0.0, 0.0) ('Twitter', 0.0, 0.0) ('am', 0.0, 0.0) ('I', 0.0, 0.0) ('only', 0.0, 0.0) ('one', 0.0, 0.0) ('problem', 0.0, 0.625)	1.5	0.057692307692307696
P2	('i', 0.0, 0.0) ('love', 0.625, 0.0) ('picture', 0.0, 0.0)	0.625	0.15625

Table 19: SentiWordNetScores for P1 and P2

In summary, the removal of noisy microblog posts can be seen as a classification problem in which there are 2 classes, noisy posts (adverts and spam) and non-noisy posts (subjective posts). The decision tree in Figure 10 represents the process of classification.

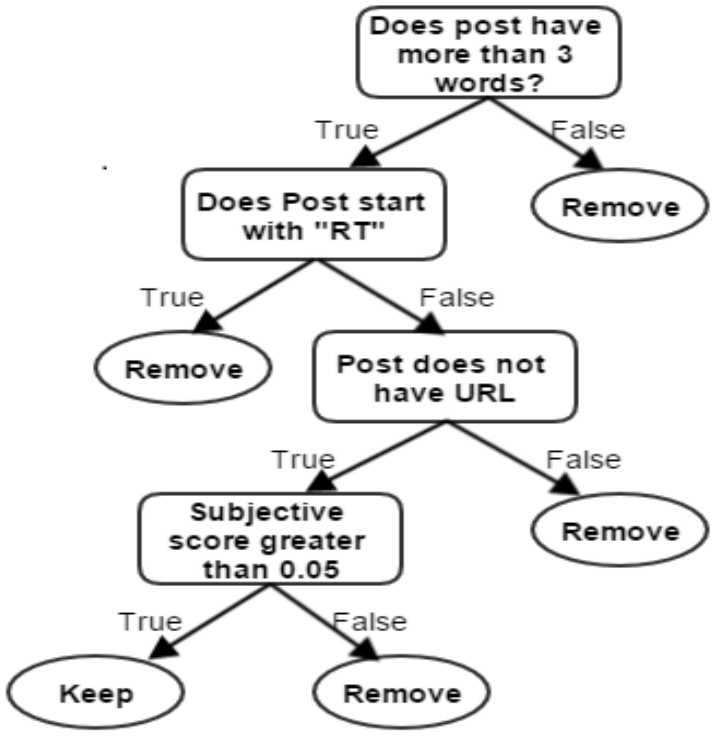


Figure 11: Binary Tree Used for Classifying Posts as Subjective

3.2 Obtaining Aspects of the Product

The aspects of the product are obtained from the posts remaining after removing all the noisy microblog posts. These posts are the subjective posts because they express an opinion on the product. An Aspect Mining Module is introduced to obtain the aspects of the product. The major steps taken in this module are:

1. Obtain the frequent nouns with a minimum support of 1%
2. Obtain the vector representation of the frequent nouns using word2vec algorithm (Mikolov et al. 2013).
3. Cluster the word vectors using a modified form of KMeans to obtain the aspects of the product.

This module is discussed in details in the next sub-section

3.2.1 Aspect Mining Module (AMM)

In this module, the aspect terms are mined from the subjective posts. The term, ***candidate aspect term*** will be used to describe the discovered aspects extracted in this module and should not be confused with the real aspects of the product because some discovered aspects are not relevant aspects of the product. The algorithm for mining the aspects is shown below:

Algorithm: to mine aspects from microblog posts about an entity (AMM)
Input: Subjective microblog posts, *SP* from subjectivity module, Language Model, *L*.
Variables:, Sim(a,b): 1- Cosine Similarity(a,b)
***e*:** product the microblog posts are about
Sim(a,b): The similarity between words a and b (i.e. 1- Cosine Similarity(a,b))
V: Vocabulary of words in our language model
Output: Candidate aspect terms
START

```
1  FOR post, p in SP DO
2      Tokenize p and remove stopwords
3      POS tag each word, w in p
4      Obtain all Noun and Plural Noun POS tags in p as nouns
5      Apply appriori algorithm on nouns (section 1.2)
6      Obtain all words with minimum support of 1% in nouns as frequent_nouns
7   FOR word, w in frequent_nouns DO:
8      IF w in V and Sim (e, w) is greater than 0.4 THEN:
9          w is a candidate aspect of product, e
10         Obtain vector representation of w
11         Add vector representation of w to the collection c_aspects
12 Select 2 arbitrary points including e in c_aspects as centers
13 Calculate the Euclidean distance (equation 3.2 below) between each w in c_aspects
   and the centers
14 Assign w to the nearest center to form a cluster, c
15 For every cluster, c obtain the center by getting the mean of all w in the cluster
16 Repeat steps 13-15 until centers have a constant value
```

STOP

Algorithm 3.3: Aspect Mining Algorithm

Tokenization

In line 2 of Algorithm 3.2 above, each post in the subjective posts is broken down into word tokens. This form of tokenization was done using Twokenizer which is a tokenizer built specifically for

tweets (Owoputi et al. 2013). Twokenizer recognizers special Twitter characters when performing tokenization which other general tokenizers such as NLTK[6] Tokenizer ignores. For example consider the microblog post below:

P1: *@WheatGeerJJ @alexiskienlen @southpawmegan @AGloverAgronomy I think I will get a iPhone6. still working on 5 twitter apps, Better Pics too http://t.co/ph0wt84ZXP*

The output of tokenizing **P1** using Twokenizer and NLTK Tokenizer is shown in Table 19 below:

Twokenizer	NLTK Tokenizer
['@WheatGeerJJ', '@alexiskienlen', '@southpawmegan', '@AGloverAgronomy', 'I', 'think', 'I', 'will', 'get', 'a', 'iPhone6', '.', 'still', 'working', 'on', '5', 'twitter', 'apps', ',', 'Better', 'Pics', 'too', 'http://t.co/ph0wt84ZXP']	['@', 'WheatGeerJJ', '@', 'alexiskienlen', '@', 'southpawmegan', '@', 'AGloverAgronomy', 'I', 'think', 'I', 'will', 'get', 'a', 'iPhone6', '.', 'still', 'working', 'on', '5', 'twitter', 'apps', ',', 'Better', 'Pics', 'too', 'http', ':', '//t.co/ph0wt84ZXP']

Table 20: Results of Tokenization of P1

From Table 3.4 we can see that usernames (words that start with a '@' symbol) are not tokenized properly with the NLTK Tokenizer but the Twokenizer, it recognises the username as a token on its own and performs the tokenization process accordingly. Also it can be seen that tokenizers not built for Tweets like NLTK Tokenizer breaks up URLs while Twokenizer recognizes the URL as a token on its own.

Removing Stopwords

As mentioned in section 1.2.2, stopwords are commonly used words that have little or no impact on the mining process. In this thesis, the list of stopwords used are NLTK stopwords and they are shown in the table below:

[6] NLTK (Natural Language Tool Kit) is a library in Python used for Text Mining

'i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', 'her', 'hers', 'herself', 'it', 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don', 'should', 'now'

Table 21: List of Stopwords in NLTK

In line 2 of the Algorithm 3.2, after each post is broken down into word tokens (tokenization) any word that belongs to the list above in table 20 is removed (removal of stopwords).

Part-of-Speech Tagging

In line 3 of Algorithm 3.2 above, each word token is assigned a part-of-speech tag. This process is called part-of-speech tagging (POS tagging) and in this thesis, this was done using NLTK POS Tagger. For example, POS tagging of the output of tokenizing **P1** above in table 3.4 will give the following:

*('@WheatGeerJJ', '**NN**'), ('@alexiskienlen', '**JJ**'), ('@southpawmegan', '**JJ**'), ('@AGloverAgronomy', '**JJ**'), ('I', '**PRP**'), ('think', '**VBP**'), ('I', '**PRP**'), ('will', '**MD**'), ('get', '**VB**'), ('a', '**DT**'), ('iPhone6', '**NNP**'), ('.', '.'), ('still', '**RB**'), ('working', '**VBG**'), ('on', '**IN**'), ('5', '**CD**'), ('twitter', '**NN**'), ('apps', '**NNS**'), (',', ','), ('Better', '**NNP**'), ('Pics', '**NNP**'), ('too', '**RB**'), ('http://t.co/ph0wt84ZXP', '**JJ**')*

The bolded above are the POS tags for the corresponding word on the left. **NN** means signifies a Noun and **JJ** signifies an adjective. The tags used in this thesis are based on the POS tagging guidelines of the Penn Treebank Project (Santorini, 1990) and the complete lists of tags and their meanings are shown below:

POS TAGS	Description
CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential there
FW	Foreign word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
NP	Proper noun, singular
NPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PP	Personal pronoun
PP$	Possessive pronoun
RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
SYM	Symbol
TO	to
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle

VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Wh-determiner
WP	Wh-pronoun
WP$	Possessive wh-pronoun
WRB	Wh-adverb

Table 22: POS Tags and their Descriptions (Santorini, 1990)

In lines 4 and 5 of Algorithm 3.2, only words that are tagged as Nouns ("**NN**") or plural nouns **(**"**NNS**") that occur at least 1% of the time in the each of the subjective posts are selected as frequent nouns. For example, if we have 10,000 subjective microblog posts and the word *picture* appears over 100 times (more than 1%) in the 10,000 microblog posts, it is selected as a frequent noun.

Building the Language Model

The purpose of the language model is to obtain the vector representations of words. Each word is represented by a 200-dimension vector in such a way that words that have similar meanings have similar vector representations. For example the vector representation of *water* will be more similar to the vector representation of *liquid* than the vector representation of *house*. The steps taken here are as follows:

i. Obtained 2 Billion tweets from the Stanford NLP Group (Pennington et al. 2014). This contains 1.2 million unique words.
ii. The corpus from the step above is used as input in the word2vec algorithm (Mikolov et al., 2013). The steps taken in this algorithm have been discussed in section 2.2.2 and it was implemented using the Genism Toolkit (Rehurek and Sojka 2010).

A language model which contains 1.2 million unique words mapped to vectors of dimension 200 each is obtained as the output.

The vector representations of the frequent noun are looked up in the language model and obtained. In lines 6-9, each frequent noun is checked to see if it is in the vocabulary of the language model (section 2.2.2 above), if the similarity between it and the product is greater than 0.4 and if it is

longer than 2 in length. The frequent nouns that do not meet all these 3 criteria are dropped. In line 9, we obtain the vector form of the remaining frequent nouns.

At this stage, each word is represented as a vector of length 200. Lines 12-16 shows an extension of KMeans clustering algorithm (section 1.2.1) with our *k* (number of clusters) set to 2. The distance measure used is Euclidean Distance and the formula is given by:

$$d_{xy} = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$ (Cover and Hart 1967) (3.2)

Where *x* and *y* are vectors that represent words.

The cluster that contains the product, is chosen. Therefore, words in that same cluster are chosen as candidate aspects.

Obtaining Relevant Aspects

To obtain the relevant aspects of a product we introduce a term called the Aspect-Product Similarity Threshold (APSM). This is the threshold at which the cosine similarity between a product and its aspect falls. From experiments, this threshold is observed to be 0.7. Candidate aspects that are above this threshold mostly competitor's products or parent companies of the product and are therefore not treated as aspects of the product. The relevant aspects are also ranked used the cosine similarity. The closer a candidate aspect is to the APSM, the higher it is ranked as an aspect of the product.

3.3 Obtaining the Opinion Polarity of the Aspects

The Aspect Opinion Mining module is used to obtain the polarity of the aspects of the products from the Tweets. In this thesis, we classify the opinion polarity as positive, negative or neutral. The algorithm for obtaining the opinion polarity of the discovered aspects is shown below:

Algorithm: to mine opinions on each aspects of an entity (AOMM)

Input: Discovered aspects, $A = \{a_1, a_2, \ldots a_n\}$ of product, e

Variables:

***e*:** entity the microblog posts are about

SP: collection of subjective microblog posts

pos_op: positive opinions; *neg_op*: negative opinions; *neutral_op*: neutral opinions

pos_words, *neg_words*: set of positive and negative words

Sim(a,b): The similarity between words a and b (i.e. 1- Cosine Similarity(a,b))

Output: Opinion polarity of each aspect term

START

1 **FOR** each a_i in A **DO:**

2 Get posts, P_{ai} that mentions a_i in SP

3 **FOR** each post, p in P_{ai} **DO**:

4 Get the nearest verbs and adjectives, V_{ai}

5 **IF** *Sim (pos_words,* V_{ai}*)* is < than *Sim (neg_words,* V_{ai}*)* **DO:**

6 Increase *pos_op* by 1

7 **ELSE IF** *Sim (pos_words,* V_{ai}*) < Sim (neg_words,* V_{ai}*)* **DO:**

8 Increase *neg_op* by 1

9 **ELSE DO:**

10 Increase *neutral_op* by 1

11 Get percentage of positive, negative and neutral opinions for aspect, a_i

12 **STOP**

ALGORITHM 3.4: Aspect Opinion Mining Algorithm

In line 4 of Algorithm 3.3 above, we get the verb or adjective to the discovered aspect in the microblog post. We then compare it with a set of predefined positive words and negative words. If the verb or adjective is more similar to the set of positive words, we classify the aspect as positive in that particular post and vice-versa. If the similarity is equal (to 4 decimal places), the aspect is classified as neutral. Also, this is toggled is a "not" is found just before the adjective or verbs. We then count the number of posts that the aspect was classified as positive, negative or neutral to get the summary of the percentage polarity of the aspect.

3.4 A Walk through Example

To demonstrate the entire workflow of our proposed framework, we would use microblog posts (tweets) downloaded from Twitter. The microblog posts are about the iPhone, hence the iPhone is our entity of interest in on which we will perform ABOM. Assuming we have a collection of 300,000 microblog posts about the iPhone, our task is to obtain the aspects of the iPhone that people are frequently talking about the overall opinion of people on each aspect. For the sake of space, Table 17 below shows 20 randomly selected microblog posts which we will use in the discussion.

SN	MICROBLOG POSTS (Tweets)
1	The back of my iphone is getting hot while charging. Is it normal? Or i did something wrong?
	@Android i cant conect my iphone 6 with the android moto 360. Help me please.
2	@_kaliblaze I don't have an iPhone so idk if you iPhone people widdit. But it's 347-893-7603
3	RT @jdbstrophy: I want Justin to do another #MusicMonday for the next 400 mondays with the 400 unreleased songs he has on his iPhone.
4	iPhone 6 are a pain for phone cases ðŸ˜, I mean why make a phone so thin & not bring out
5	@bigbunnny told u i needed a iphone charger nf
6	How to Make Sharing from Your iPhone Work Better for You! #iPhone #Tips http://t.co/VwELP5UITY via @wonderoftech
7	RT @CechyNandos: If Benzema doesn't join Arsenal, I will buy every person who RT'S an iPhone 6. That's how confident I am
8	Bing for iPhone picks up Interests and News, gains private searching, Apple Watch app and more http://t.co/A0xX0VLOHl
9	Definitely need to get this iPhone screen fixed!!
10	I'm ready for Apple to drop a new iPhone. I'm over this 6 plus ... I want a smaller phone
11	RT @ivannnarod: Just want @apple to drop a new iPhone so I can upgrade
12	Gonna buy two more #iphone chargers plug one in for home! Plug one in for work.. ðŸ˜’

13	does anyone have an extra iPhone charger ðŸ˜©
14	lol @JohnLegere was talking about the iPhone 6s on his periscope
15	I Hate How iPhone Chargers Werent Made To Last Long.
16	My new iPhone battery sucks so bad
17	This iPhone hack will make your battery last for a whole week: http://t.co/8IAOO5gDV6 via @AOL
18	RT @QuazzXo: iPhone battery percentage is ðŸ˜· to me
19	RT @ElaineBaldwin86: If the inventor of the iPhone battery ever ends up on life support in a hospital, I hope the back up power sourc
20	lmao at my iphone cutting off and not cutting back on @ 89% battery

Table 23: Sample Microblog Posts

The MAM algorithm class the Pre-processing module.

STEP 1: The posts are run through the preprocessing modules to "clean" them up. The posts that contain a URL and (or) start with "RT" are discarded and foreign characters and symbols are removed. We obtain the following results in table 3.8 below after preprocessing. It can be noticed that eight posts were dropped.

SN	Pre-Processed Posts
1	i cant conect my iphone 6 with the android moto 360. Help me please.
2	iPhone 6 are a pain for phone cases I mean why make a phone so thin & ; not bring out
3	Definitely need to get this iPhone screen fixed
4	lol was talking about the iPhone 6s on his periscope
5	I Hate How iPhone Chargers Werent Made To Last Long
6	Gonna buy two more #iphone chargers plug one in for home Plug one in for work..
7	I'm ready for Apple to drop a new iPhone. I'm over this 6 plus ... I want a smaller phone
8	told u i needed a iphone charger nf

9	kaliblaze I don't have an iPhone so idk if you iPhone people widdit. But it's 347-893-7603
10	does anyone have an extra iPhone charger
11	lmao at my iphone cutting off and not cutting back on 89% battery
12	The back of my iphone is getting hot while charging. Is it normal? Or i did something wrong?

Table 24: Output of Preprocessed Sample Microblog Posts

STEP 2: Obtain the subjective posts by running the preprocessed posts through the subjectivity module using Algorithm 3.2. From the results below in table 24, it is observed that posts 6 and 8 in Table 23 above were deemed not subjective by the subjective module, hence, they were dropped.

SN	Subjective Posts
1	i cant conect my iphone 6 with the android moto 360. Help me please.
2	iPhone 6 are a pain for phone cases I mean why make a phone so thin & ; not bring out
3	Definitely need to get this iPhone screen fixed
4	lol was talking about the iPhone 6s on his periscope
5	I Hate How iPhone Chargers Werent Made To Last Long.
6	I'm ready for Apple to drop a new iPhone. I'm over this 6 plus ... I want a smaller phone
7	told u i needed a iphone charger nf
8	My new iPhone battery sucks so bad
9	lmao at my iphone cutting off and not cutting back on 89% battery
10	The back of my iphone is getting hot while charging. Is it normal? Or i did something wrong?

Table 25: Sample Subjective Posts

The next steps take place in the Aspect Mining Module (AMM).

STEP 3: We tokenize and remove stopwords from the subjective posts. The result of this is shown in table 25 below:

SN	Subjective Posts
1	'cant', 'conect', 'iphone', '6', 'android', 'moto', '360', '.', 'help', 'please', '.'
2	'iphone', '6', 'pain', 'phone', 'cases', 'i', 'mean', 'make', 'phone', 'thin', '&', ';', 'bring'
3	definitely', 'need', 'get', 'iphone', 'screen', 'fixed'
4	'lol', 'talking', 'iphone', '6s', 'periscope'
5	'i', 'hate', 'how', 'iphone', 'chargers', 'werent', 'made', 'to', 'last', 'long', '.'
6	"i'm", 'ready', 'apple', 'drop', 'new', 'iphone', '.', "i'm", '6', 'plus', '...', 'i', 'want', 'smaller', 'phone'
7	'told', 'u', 'needed', 'iphone', 'charger', 'nf'
8	'my', 'new', 'iphone', 'battery', 'sucks', 'bad'
9	'lmao', 'iphone', 'cutting', 'cutting', 'back', '89%', 'battery'
10	'the', 'back', 'iphone', 'getting', 'hot', 'charging', '.', 'is', 'normal', '?', 'or', 'something', 'wrong', '?'

Table 26: Word Tokens for each of the Sample Subjective Posts

STEP 4: Each of the word tokens in the subjective posts are assigned a part-of-speech tag (POS Tag) and the Nouns and Plural Nouns are chosen. Table 26 below shows all the nouns and plural nouns in the sample subjective posts in Table 25 above and their frequency of occurrence in bold.

'iphone': **10**, 'phone': **3**, 'battery': **2**, "i'm": **2**, 'help': **1**, 'back': **1**, 'cant': **1**, 'periscope': **1**, 'nf': **1**, 'lol': **1**, 'chargers': **1**, 'hot': **1**, 'pain': **1**, 'get': **1**, 'screen': **1**, 'cutting': **1**, 'werent': **1**, 'cases': **1**, 'android': **1**, 'lmao': **1**, 'charger': **1**, 'drop': **1**, 'sucks': **1**, 'plus': **1**, 'something': **1**

Table 27: Nouns in the Subjective Posts and their respective Frequencies

It should be noted that with a larger dataset, we get more words and higher frequencies. Also, it is at this step that most of related studies stop and take the most frequent words based on a set minimum support as aspects of the entity without considering if they are semantically related to the entity (iphone in this case).

STEP 5: In this step, we prune off the list of nouns by selecting only nouns that occur with a minimum support of 1% in the subjective posts as our frequent nouns. For example, if we have 100,000 subjective posts, any noun that occurs more than 100 times is selected. Since we have only 10 subjective posts in this example, we will take all the nouns in Table 21 above as our frequent nouns. We then drop every frequent noun shorter than 2 letters (*i`m* and *nf* in this example) and we then check how semantically related each of the frequent nouns are to the product under consideration (i.e the iPhone). The semantic similarity between each frequent noun and the product is shown in Table 27 below:

SN	Frequent Nouns	Similarity with Product
1	help	0.3306
2	battery	0.4642
3	back	0.4164
4	cant	0.3610
5	periscope	-0.0737
6	lol	0.4337
7	chargers	0.3430
8	hot	0.3497
9	iphone	1.0000
10	pain	0.2353
11	get	0.4290
12	screen	0.5685

13	phone	0.7158
14	cutting	0.2401
15	werent	0.0388
16	cases	0.4525
17	android	0.7785
18	lmao	0.3324
19	charger	0.5584
20	drop	0.3561
21	sucks	0.3412
22	plus	0.3960

Table 28: Similarity Score between Frequent Nouns and Entity

From Table 22 above, we can see that a frequent noun like *iphone* has a similarity score of 1.0 with the entity because it is the same as the entity. Also, words that are not related to the entity at all have a lower score than words related to the entity. For example, *periscope* has a score of -0.0737 while *screen* has a similarity score of 0.5685. We prune the frequent nouns list by selecting only frequent nouns with similarity score of over 0.4. Thus our frequent noun list becomes: {*battery, back, lol, iphone, get, screen, phone, cases, Android, charger*}.

STEP 6: We extend KMeans clustering algorithm to this pruned frequent noun list to divide them into two clusters. Figure 11 below shows the output of clustering the pruned frequent nouns list.

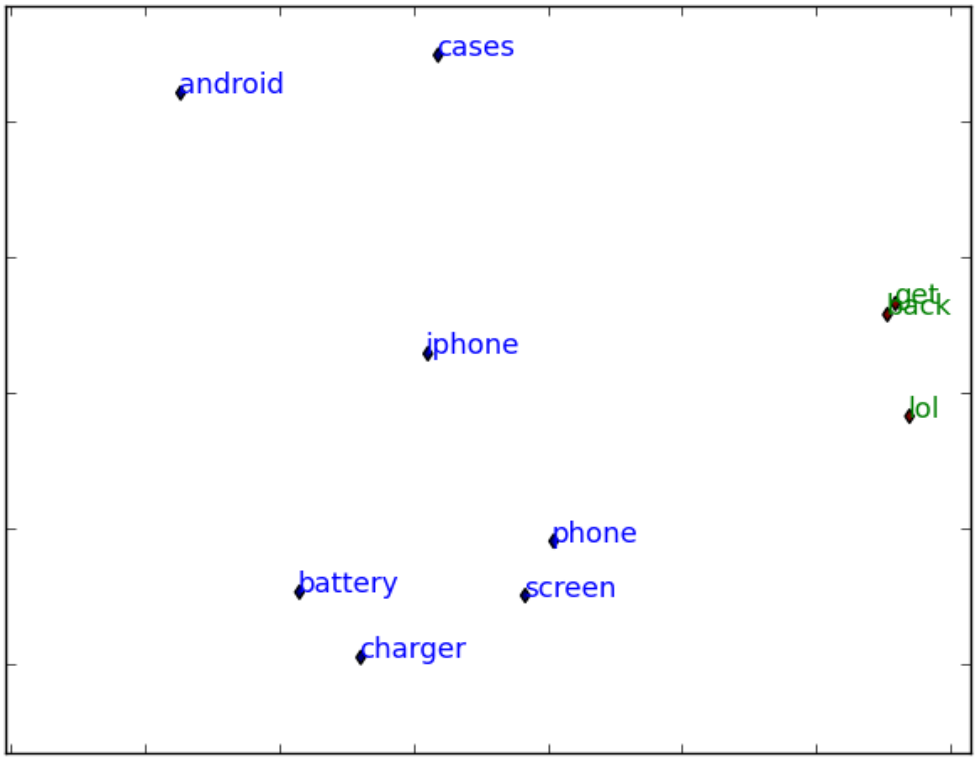


Figure 12: Result of Clustering the Pruned Frequent List

From the figure above, we can see clearly the two clusters formed:

Cluster 1 = {*get, back, lol*}

Cluster 2 = {*android, cases, iphone, phone, screen, battery, charger*}

We select the cluster that has the entity term (iphone in this case). Hence, items in Cluster 2 are selected as our candidate aspects.

STEP 7: To obtain the relevant aspects of the iphone, we drop any word in the candidate aspect that do not fall below the Aspect-Product Similarity Threshold. Using the similarity scores in Table 22 above, the words *iphone, android* and *phone* are pruned. Therefore, the Aspect Mining Module gives the following as the aspects of the entity, iPhone: {*screen, charger, battery, cases*}. These are ranked according to their similarity with the iphone as seen in Table 22 above.

STEP 8: Using the discovered aspects, the next step is to get the opinions of people on each of these discovered aspect to know if they are positive or negative by running them through the Aspect Opinion Mining (AOM) module. We look up the subjective posts (Table 19 above) to get the posts in which these discovered aspects were mentioned. The posts are shown below in Table 28:

SN	Aspect	Microblog Post
1	cases	iPhone 6 are a pain for phone **cases** I mean why make a phone so thin & ; not bring out
2	screen	Definitely need to get this iPhone **screen** fixed
3	battery	My new iPhone **battery** sucks so bad
4	battery	lmao at my iphone cutting off and not cutting back on 89% **battery**
5	charger	told u i needed a iphone **charger** nf

Table 29: Aspects and the Posts in which they appear in

From the AOM module, we get a summary of the opinions of each aspect which is the final output of the system. In this example, the summary is given below:

SN	Aspect of Iphone	Opinion
1	cases	Negative (100%); Positive (0%); Neutral (0%)
2	screen	Negative (100%); Positive (0%); Neutral (0%)
3	battery	Negative (100%); Positive (0%); Neutral (0%)
4	charger	Negative (100%); Positive (0%); Neutral (0%)

Table 30: Final Output of the System

CHAPTER 4: EXPERIMENTS AND ANALYSIS

This section discusses the implementation and experiments performed to evaluate our proposed system's effectiveness in terms of precision and accuracy in mining the aspects of a product

4.1 Datasets

In this thesis we used 500,000 tweets from 4 products and brands from different as our text corpus. The products are *iphone, starbucks, xbox* and *sony*. Iphone and Xbox were chosen because they are one of the most talked about products on twitter and Starbucks and Sony are easily recognizable brands. We obtained English tweets from Twitter over a period of 1 month[7].

4.2 Experiment Setup

We test our system against other Aspect-based Opinion Miners like TAC (Lek and Poo, 2013) and Frequent Noun (FN) based systems like TF-IDF (Spina et. al 2013). We implement this systems and get the aspects of the products from collected Microblog Posts about the product which were over 300,000 tweets. We give three human judges who know the products and have used them before to rate how relevant the aspects are to the products on a scale of 0-3. The rating scale is as follows: 0 (not a relevant aspect of the product), 1 (vaguely relevant to the product), 2 (slightly relevant aspect of the product), 3 (relevant aspect of the product). Table 31 below shows the score that each of the judges gave for each of the aspects of the iphone produced by the 3 different systems. The judges did not know which system produced what output. They were just asked to give the score.

[7] Spetember 9 – October 9, 2015

	Judge1	Judge2	Judge3
verizon	0	1	1
boyfriend	0	0	0
mrcocoyam	0	0	0
notifications	0	2	2
upgrade	0	2	1
apple	0	0	0
galaxy	0	0	0
android	0	0	0
bluetooth	3	2	3
smartphone	0	0	0
samsung	0	0	0
#androidapps	2	0	0
ios	3	3	3
wallet	2	0	0
#deals	2	0	0
gana	0	0	0
ipod	0	0	0
ipad	0	0	0
#iphonegames	1	1	1
@jasonortiaga	0	0	0
app	3	3	3
phones	0	0	0
screen	3	3	3
case	3	2	3
charger	3	3	3
camera	3	3	3
cell	3	3	3
verizon	0	1	1
users	0	0	2
store	3	3	3
sprint	0	1	1
battery	3	3	3

ebay	0	0	0
tech	0	1	0
gold	3	0	3
price	3	3	3
launch	0	0	0
sale	2	2	2
brand	3	0	3
silver	3	0	3
case	3	2	3
apple	0	0	0
#iphone	0	1	1
gold	3	0	3
get	2	3	3
app	3	3	3
phone	0	0	0
apps	0	0	0
smartphone	0	0	0
link	2	2	1
ios	3	3	3
video	3	2	3
music	0	0	0
size	0	0	0
plus	0	0	0
check	0	0	0
shell	0	0	0
funny	0	0	0
retro	0	0	0
vintage	0	0	0

Table 31: Judges' Score of the Aspects of the Iphone Dataset Produced by the 3 Systems

We compute the agreement of the judges using the Cohen's and Fleiss' Kappa (Fleiss, Cohen and Everitt 1969) to see if the rating of the 3 judges are in agreement. The kappa value is used to measure how consistent the ratings giving the judges are (Spina et. al 2013) and the formula to calculate it is given as:

$$\kappa = \frac{Po-Pc}{1-Pc} \quad \text{(Fleiss, Cohen and Everitt 1969)}$$

Where $P_o - P_c$ gives the degree of agreement achieved above chance and $1-P_c$ gives the degree of agreement attainable by chance. The formula to calculate P_c is given by

$$P_c = \sum_{j=1}^{k} p_j^2 \quad \text{(Fleiss, Cohen and Everitt 1969)}$$

Where *k* is the number of ratings that can be given (in this case 4 ratings can be given) and the formula to calculate p_j is given as:

$$P_{j=\frac{1}{Nn}\sum_{i=1}^{N} n_{ij}} \text{ (Fleiss, Cohen and Everitt 1969)}$$

Where *N* is the number of aspects that were scored, *n* the number of judges and n_{ij} is the number of judges who assigned *i*-th aspect to the *j*-th rating.

And the formula to calculate Po is given as:

$$P_o = \frac{1}{Nn(n-1)}\left(\sum_{i=1}^{N}\sum_{j=1}^{k} n_{ij}^2 - n\right) \text{ (Fleiss, Cohen and Everitt 1969)}$$

A kappa value of 0 shows no agreement between the judges and a kappa value of 1 shows a strong agreement between the judges. The highest value for a Kappa value is 1 and this shows that all the judges gave the same ratings to all aspects. Kappa values below 0 show no agreement between the judges. Kappa scores from 0.6 up indicate a good enough agreement among the judges to validate their decisions (Spina et. al 2011). Table 32 below shows the kappa scores of how the judges rated the aspects in our experiment for each dataset. All Kappa scores are above 0.6 hence the scoring of the judges have been validated.

Dataset	Kappa Score	Interpretation
Iphone	0.6073	Moderate Agreement
Starbucks	0.6517	Moderate Agreement
Sony	0.6107	Moderate Agreement
Xbox	0.6271	Moderate Agreement

Table 32: Kappa Scores of Judges' Ratings of the 4 Products

The results above show a strong agreement in how the judges scored the aspects of the products.

4.3 Evaluation Metrics

Since the study is to get the most relevant aspects, we evaluate the top 20 aspects of each of the systems and we use the following evaluation metrics:

Precision – This measures amount of relevant aspects that were among the top 20 aspects. The relevant aspects are those aspects given a score of 3 by the judges.

Weighted Precision (Sakai 2007) – This measures the relevancy of the aspect to the product based on the scoring of the judges. Let *R(a*) be the average rating given by the judges for each aspect, *a*, the weighted precision is given as:

Weighted Precision = $\frac{\sum_1^K R(a)}{K}$ (Sakai 2007).

Where *K* is 20 since we ae considering the top 20 aspects. Note that the value of the weighted precision lies between 0 and 3 with 3 being the most relevant and 0 being not relevant.

It should be noted that metrics such as Recall and F1-Measure was not used in this case as it will be impractical to determine all the aspects in each tweet in the dataset manually so as to get the true positives. Hence we only use metrices that do not require knowing all the true positives in the datasets.

4.4 Results

The average weighted precision was got using our proposed solution (MAM), TAC which uses the PMI method (Lek and Poo 2013) and the just by counting the top 20 most frequent nouns (FN) for each which is serves as the baseline (Spina et al. 2012). The results were averaged across the 4 products and are shown in Figure 13 below:

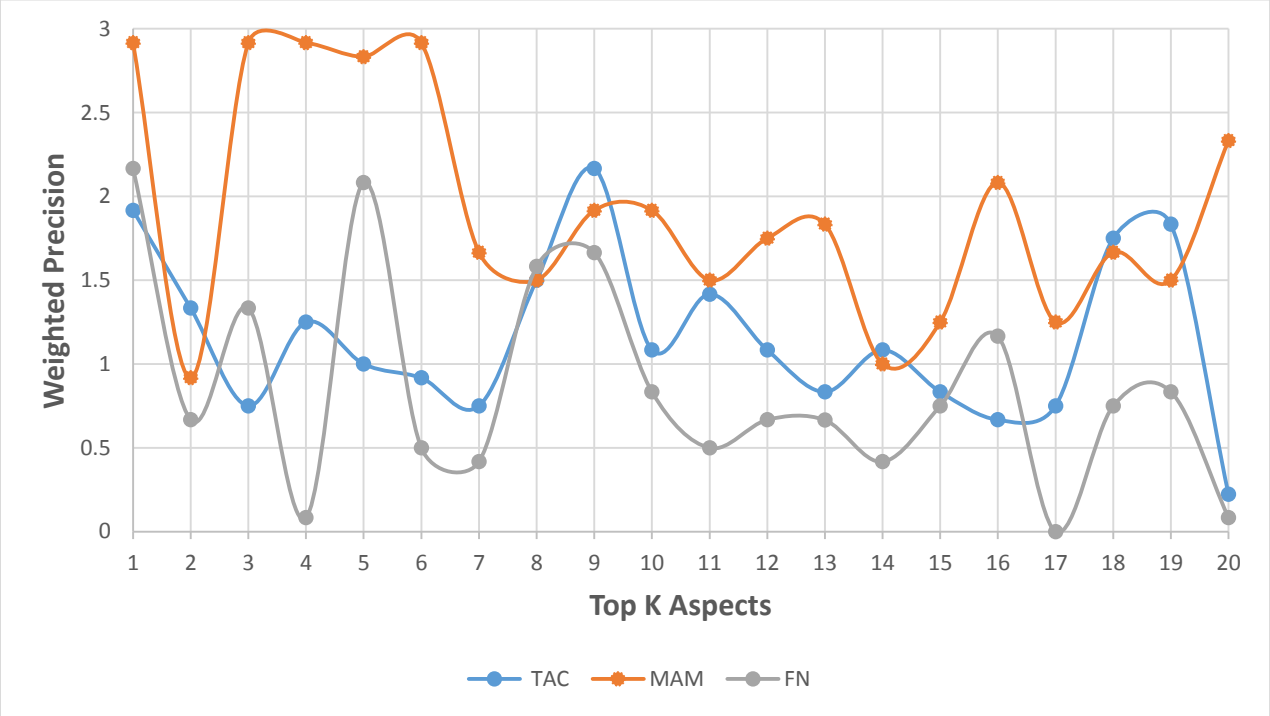


Figure 13: Average Weighted Precision across the Four Products

It can be seen that all the methods did quite well in choosing a quite relevant aspect of the products as the first aspect. The following can also be observed from the figure above:

1. 70% of the aspects predicted by FN had a weighted precision score of less than 1. This shows that just picking the most frequent nouns that occur about a product on Microblogs gives a poor result in determining the aspect of the product. This can be attributed to the noisy nature of Microblogs.
2. MAM shows a very accuracy in predicting relevant top 6 aspects and the accuracy drops after then. Only one of the predicted aspects had a weighted precision that was less than 1

3. MAM outperforms TAC on the average in predicting the top 20 aspects across the 4 products.
4. Although TAC showed to be more stable from the figure, the accuracy of prediction of aspects is more important than the stability of prediction

Furthermore, defining "relevant aspects" of products as aspects that were given a perfect score of 3 by the judges, the number of relevant aspects obtained for each dataset is shown in the figure below:

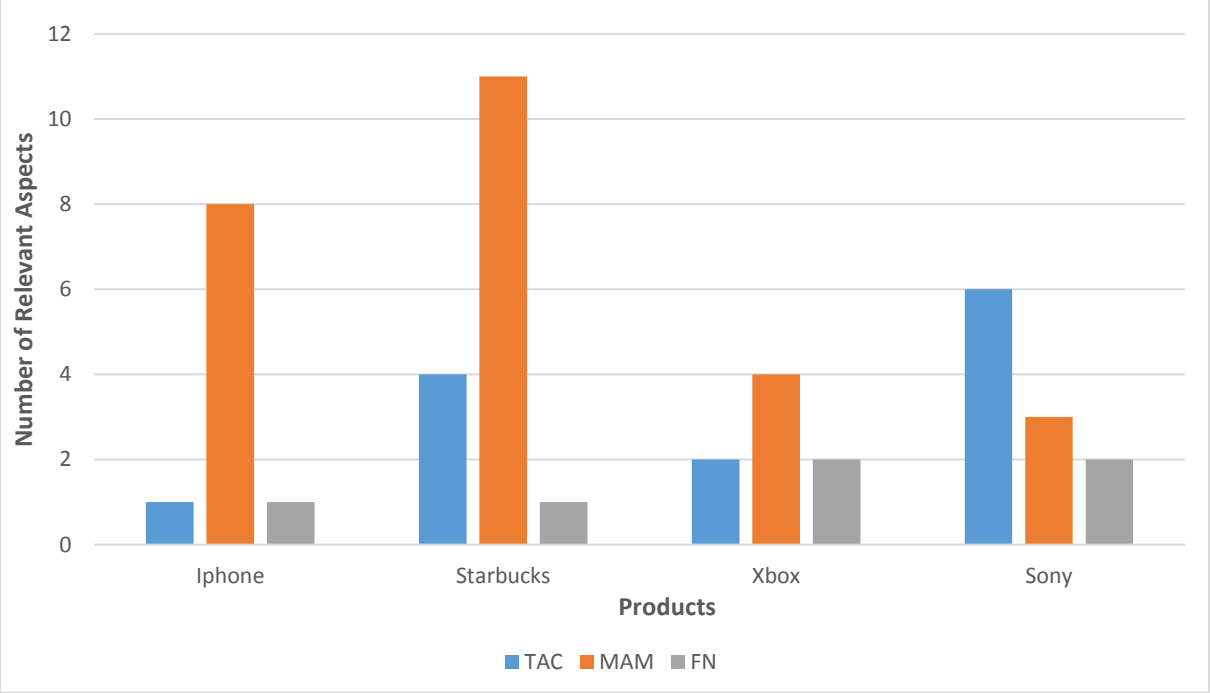


Figure 14: Number of Relevant Aspects Obtained for Each Dataset

It can be seen that MAM performed poorly with the Sony dataset. This can be attributed to the heterogeneity of the words in the Sony dataset as Sony is a large conglomerate with different and dissimilar products and services. Therefore, MAM which relies on clustering based on sematic similarity did not perform well.

CHAPTER 5: CONCLUSION AND FUTURE WORKS

In this thesis, we proposed a method, Microblog Aspect Miner (MAM) for mining aspects of products from microblogs by dealing with the noisy posts in microblogs and using a clustering approach to get the relevant aspects of a product. Previous research have considered this problem but their accuracy in determining the aspects of a product is affected by spam posts and also they do not explore the semantic similarity. MAM uses a preprocess module which calculates subjectivity of posts to remove the spam posts and uses a clustering approach to explore the product-aspect semantic relationship. Experimental results show that the proposed technique performs better in terms of accuracy of getting the relevant aspects of a product. Furthermore obtaining the opinions on these discovered aspects can give business owners an insight into what customers think of their business. This helps in business intelligence and decision making as this will answer some questions like “what part of my product do customers like”, “what part of my competitors’ products don’t they like?”

Below are some interesting extensions of this study and some avenues to explore for future works:

1. This work only considered Twitter posts in English. ABOM in Microblogs in other languages will provide a more global insight into what customers think of a product
2. Some aspects of a product are more than one word, (e.g. hard disk) and this work cannot handle such situations because the language model used cannot create vector representations for multiple word aspects. Techniques to cluster multi-word aspects are needed for cases like this.
3. Aggregation of all posts on Microblogs, Blogs, News Articles and Product Reviews to perform ABOM.
4. A system to rate different aspects of similar products using posts from twitter to aid customers in making better purchasing decisions.
5. Introducing a more rigorous approach by clearly defining subjective and objective features of microblog posts in the subjectivity classifier to improve the accuracy.

REFERENCES

Aggarwal, Charu C., and ChengXiang Zhai, eds. 2012. *Mining Text Data*. Boston, MA: Springer US. http://link.springer.com/10.1007/978-1-4614-3223-4.

Agrawal, Rakesh, and Ramakrishnan Srikant, others. 1994. "Fast Algorithms for Mining Association Rules." In *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, 1215:487–99. https://www.it.uu.se/edu/course/homepage/infoutv/ht08/vldb94_rj.pdf.

Ahmed, Sabbir, and C. I. Ezeife. 2013. "Discovering Influential Nodes from Trust Network." In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, 121–28. ACM. http://dl.acm.org/citation.cfm?id=2480389.

Asur, Sitaram, and Bernardo Huberman, others. 2010. "Predicting the Future with Social Media." In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on*, 1:492–99. IEEE. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5616710.

Baccianella, Stefano, Andrea Esuli, and Fabrizio Sebastiani. 2010. "SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining." In *LREC*, 10:2200–2204. http://www.researchgate.net/profile/Fabrizio_Sebastiani/publication/220746537_SentiWordNet_3.0_An_Enhanced_Lexical_Resource_for_Sentiment_Analysis_and_Opinion_Mining/links/545fbcc40cf27487b450aa21.pdf.

Bahrainian, Seyed-Ali, and Andreas Dengel. 2013. "Sentiment Analysis and Summarization of Twitter Data." In *Computational Science and Engineering (CSE)*, 227–34. IEEE. doi:10.1109/CSE.2013.44.

Barbier, Geoffrey, Zhuo Feng, Pritam Gundecha, and Huan Liu. 2013. "Provenance Data in Social Media." *Synthesis Lectures on Data Mining and Knowledge Discovery* 4 (1): 1–84. doi:10.2200/S00496ED1V01Y201304DMK007.

Barbosa, Luciano, and Junlan Feng. 2010. "Robust Sentiment Detection on Twitter from Biased and Noisy Data." In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, 36–44. Association for Computational Linguistics. http://dl.acm.org/citation.cfm?id=1944571.

Bengio, Yoshua, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. "A Neural Probabilistic Language Model." *The Journal of Machine Learning Research* 3: 1137–55.

Blei, David M., Andrew Y. Ng, and Michael I. Jordan. 2003. “Latent Dirichlet Allocation.” *The Journal of Machine Learning Research* 3: 993–1022.

Cambria, Erik, Bjorn Schuller, Yunqing Xia, and Catherine Havasi. 2013. “New Avenues in Opinion Mining and Sentiment Analysis.” *IEEE Intelligent Systems* 28 (2): 15–21.

Chen, Bi, Leilei Zhu, Daniel Kifer, and Dongwon Lee. 2010. “What Is an Opinion About? Exploring Political Standpoints Using Opinion Scoring Model.” In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*. Association for the Advancement of Artificial Intelligence. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.193.4589&rep=rep1&type=pdf.

Christopher D. Manning, Pennington, Jeffrey, and Richard Socher. 2014. “GloVe: Global Vectors for Word Representation.” In *Proceedings of the Empiricial Methods in Natural Language Processing*, 12:1532–43. http://www.emnlp2014.org/papers/pdf/EMNLP2014162.pdf.

Cortes, Corinna, and Vladimir Vapnik. 1995. “Support-Vector Networks.” *Machine Learning* 20 (3): 273–97.

Cover, Thomas M., and Peter E. Hart. 1967. “Nearest Neighbor Pattern Classification.” *Information Theory, IEEE Transactions on* 13 (1): 21–27.

Das, Abhimanyu, and Anitha Kannan. 2014. “Discovering Topical Aspects in Microblogs.” In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 860–71. Dublin, Ireland. http://research-srv.microsoft.com/pubs/217954/Aspects_COLING.pdf.

Dave, Kushal, Steve Lawrence, and David M. Pennock. 2003. “Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews.” In *Proceedings of the 12th International Conference on World Wide Web*, 519–28. ACM. http://dl.acm.org/citation.cfm?id=775226.

Diao, Qiming, Minghui Qiu, Chao-Yuan Wu, Alexander J. Smola, Jing Jiang, and Chong Wang. 2014. “Jointly Modeling Aspects, Ratings and Sentiments for Movie Recommendation (JMARS).” In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 193–202. ACM. http://dl.acm.org/citation.cfm?id=2623758.

Esuli, Andrea., and Sebastiani, Fabrizio. 2006). Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of LREC* (Vol. 6, pp. 417-422).

Feldman, Ronen. 2013. “Techniques and Applications for Sentiment Analysis.” *Communications of the ACM* 56 (4): 82–89.

Feldman, Ronen, and James Sanger. 2007. *The Text Mining Handbook Advanced Approaches in Analyzing Unstructured Data*. Cambridge; New York: Cambridge University Press. http://www.books24x7.com/marc.asp?bookid=23164.

Fleiss, Joseph L., Jacob Cohen, and B. S. Everitt. 1969. "Large sample standard errors of kappa and weighted kappa." *Psychological Bulletin* 72, no. 5: 323.

Go, Alec, Richa Bhayani, and Lei Huang. 2009. "Twitter Sentiment Classification Using Distant Supervision." *CS224N Project Report, Stanford*, 1–12.

Grobelnik, Marko, Dunja Mladenic, and Natasa Milic-Frayling. 2000. "Text Mining as Integration of Several Related Research Areas: Report on KDD's Workshop on Text Mining 2000." *ACM SIGKDD Explorations Newsletter* 2 (2): 99–102.

Gundecha, Pritam, and Huan Liu. 2012. "Mining Social Media: A Brief Introduction." In *2012 TutORials in Operations Research*, 1–17. INFORMS. https://www.informs.org/Pubs/Tutorials-in-OR/2012-TutORials-in-Operations-Research-ONLINE/Chapter-1.

Han, Jiawei, Micheline Kamber, and Jian Pei. 2012. *Data Mining: Concepts and Techniques*. 3. ed. The Morgan Kaufmann Series in Data Management Systems. Amsterdam: Elsevier/Morgan Kaufmann.

Hatzivassiloglou, Vasileios, and Janyce M. Wiebe. 2000. "Effects of Adjective Orientation and Gradability on Sentence Subjectivity." In *Proceedings of the 18th Conference on Computational Linguistics-Volume 1*, 299–305. Association for Computational Linguistics. http://dl.acm.org/citation.cfm?id=990864.

Hong, Yancheng, and Steven Skiena. 2010. "The Wisdom of Bookies? Sentiment Analysis vs. the NFL Point Spread." In *Proceedings of the International Conference on Weblogs and Social Media (icWSm-2010)*. Citeseer. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.368.6683&rep=rep1&type=pdf .

Hu, Minqing, and Bing Liu. 2004a. "Mining and Summarizing Customer Reviews." In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 168–77. ACM. http://dl.acm.org/citation.cfm?id=1014073.

Hu, Minqing, and Bing Liu. 2004b. "Mining Opinion Features in Customer Reviews." In *AAAI*, 4:755–60. http://www.aaai.org/Papers/AAAI/2004/AAAI04-119.pdf.

Jakob, Niklas, and Iryna Gurevych. 2010. "Extracting Opinion Targets in a Single-and Cross-Domain Setting with Conditional Random Fields." In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, 1035–45. Association for Computational Linguistics. http://dl.acm.org/citation.cfm?id=1870759.

Jansen, Bernard J., Mimi Zhang, Kate Sobel, and Abdur Chowdury. 2009. “Twitter Power: Tweets as Electronic Word of Mouth.” *Journal of the American Society for Information Science and Technology* 60 (11): 2169–88. doi:10.1002/asi.21149.

Kaplan, Andreas M., and Michael Haenlein. 2010. “Users of the World, Unite! The Challenges and Opportunities of Social Media.” *Business Horizons* 53 (1): 59–68. doi:10.1016/j.bushor.2009.09.003.

Lek, Hsiang Hui, and Danny C.C. Poo. 2013. “Aspect-Based Twitter Sentiment Classification.” In *2013 IEEE 25th International Conference on Tools with Artificial Intelligence*, 366–73. Washington: IEEE. doi:10.1109/ICTAI.2013.62.

Lim, Kar Wai, and Wray Buntine. 2014. “Twitter Opinion Topic Model: Extracting Product Opinions from Tweets by Leveraging Hashtags and Sentiment Lexicon.” In , 1319–28. ACM Press. doi:10.1145/2661829.2662005.

Li, Yung-Ming, and Tsung-Ying Li. 2013. “Deriving Market Intelligence from Microblogs.” *Decision Support Systems* 55 (1): 206–17. doi:10.1016/j.dss.2013.01.023.

Manning, Christopher D, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. New York: Cambridge University Press. http://dx.doi.org/10.1017/CBO9780511809071.

MartíNez-CáMara, Eugenio, M. Teresa MartíN-Valdivia, L. Alfonso UreñA-LóPez, and A Rturo Montejo-RáEz. 2014. “Sentiment Analysis in Twitter.” *Natural Language Engineering* 20 (01): 1–28. doi:10.1017/S1351324912000332.

McCallum, Andrew, and Kamal Nigam, others. 1998. “A Comparison of Event Models for Naive Bayes Text Classification.” In *AAAI-98 Workshop on Learning for Text Categorization*, 752:41–48. Citeseer. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.65.9324&rep=rep1&type=pdf.

Mei, Qiaozhu, Xu Ling, Matthew Wondra, Hang Su, and ChengXiang Zhai. 2007. “Topic Sentiment Mixture: Modeling Facets and Opinions in Weblogs.” In *Proceedings of the 16th International Conference on World Wide Web*, 171–80. ACM. http://dl.acm.org/citation.cfm?id=1242596.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013. “Distributed Representations of Words and Phrases and Their Compositionality.” In *Advances in Neural Information Processing Systems*, 3111–19. http://papers.nips.cc/paper/5021-di.

Milstein, Sarah, Ben Lorica, Roger Magoulas, Gregor Hochmuth, Abdur Chowdhury, and Tim O’Reilly. 2008. *Twitter and the Micro-Messaging Revolution: Communication, Connections, and Immediacy--140 Characters at a Time.* O’Reilly Media, Incorporated.

Miner, Gary D., ed. 2012. *Practical Text Mining and Statistical Analysis for Non-Structured Text Data Applications*. Waltham, Mass.: Elsevier, Acad. Press.

Moghaddam, Samaneh, and Martin Ester. 2010. “Opinion Digger: An Unsupervised Opinion Miner from Unstructured Product Reviews.” In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, 1825–28. ACM. http://dl.acm.org/citation.cfm?id=1871739.

Moghaddam, Samaneh, and Martin Ester. 2012. “Aspect-Based Opinion Mining from Product Reviews.” In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1184–1184. ACM. http://dl.acm.org/citation.cfm?id=2348533.

Morinaga, Satoshi, Kenji Yamanishi, Kenji Tateishi, and Toshikazu Fukushima. 2002. “Mining Product Reputations on the Web.” In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 341–49. ACM. http://dl.acm.org/citation.cfm?id=775098.

Mumu, Tamanna S., and Christie I. Ezeife. 2014. “Discovering Community Preference Influence Network by Social Network Opinion Posts Mining.” In *Data Warehousing and Knowledge Discovery*, 136–45. Springer. http://link.springer.com/chapter/10.1007/978-3-319-10160-6_13.

Nakov, Preslav, Sara Rosenthal, Zornitsa Kozareva, Veselin Stoyanov, Alan Ritter, and Theresa Wilson. 2013. “SemEval-2013 Task 2: Sentiment Analysis in Twitter.” In *Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, 2:312. Atlanta, Georgia: Association for Computational Linguistics. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.307.9968&rep=rep1&type=pdf#page=348.

Niu, Zhen, Zelong Yin, and Xiangyu Kong. 2012. “Sentiment Classification for Microblog by Machine Learning.” In , 286–89. IEEE. doi:10.1109/ICCIS.2012.276.

O’Connor, Brendan, Ramnath Balasubramanyan, Bryan R. Routledge, and Noah A. Smith. 2010. “From Tweets to Polls: Linking Text Sentiment to Public Opinion Time Series.” In *Proceedings of the Fourth International AAAI Conference on Weblogs and Social Media*. Association for the Advancement of Artificial Intelligence. http://www.aaai.org/ocs/index.php/ICWSM/ICWSM10/paper/viewFile/1536/1842/.

Owoputi, Olutobi, Brendan O’Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A. Smith. 2013. “Improved Part-of-Speech Tagging for Online Conversational Text with Word Clusters.” In . Association for Computational Linguistics. http://repository.cmu.edu/lti/47/.

Pak, Alexander, and Patrick Paroubek. 2010. “Twitter as a Corpus for Sentiment Analysis and Opinion Mining.” In *Proceedings of the Seventh Conference on International Language Resources and Evaluation*, 10:1320–26. http://incc-tps.googlecode.com/svn/trunk/TPFinal/bibliografia/Pak%20and%20Paroubek%20(2010).

%20Twitter%20as%20a%20Corpus%20for%20Sentiment%20Analysis%20and%20Opinion%20Mining.pdf.

Pang, Bo, 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*, 79–86. Association for Computational Linguistics. http://dl.acm.org/citation.cfm?id=1118704.

Pavlopoulos, Ioannis. 2014. “Aspect Based Sentiment Analysis.” PH.D. THESIS, ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS. http://nlp.cs.aueb.gr/theses/ipavlopoulos-thesis.pdf.

Perman, Mihael, Jim Pitman, and Marc Yor. 1992. “Size-Biased Sampling of Poisson Point Processes and Excursions.” *Probability Theory and Related Fields* 92 (1): 21–39.

Popescu, Ana-Maria, and Orena Etzioni. 2005. “Extracting Product Features and Opinions from Reviews.” In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, 339–46. Vancouver: Association for Computational Linguistics.

Řehůřek, Radim, and Petr Sojka. 2011. "Gensim—Statistical Semantics in Python." .

Riloff, Ellen, Siddharth Patwardhan, and Janyce Wiebe. 2006. “Feature Subsumption for Opinion Analysis.” In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, 440–48. Association for Computational Linguistics. http://dl.acm.org/citation.cfm?id=1610137.

Ritter, Alan, Sam Clark, and Oren Etzioni, others. 2011. “Named Entity Recognition in Tweets: An Experimental Study.” In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 1524–34. Association for Computational Linguistics. http://dl.acm.org/citation.cfm?id=2145595.

Sadikov, Eldar, Aditya Parameswaran, and Petros Venetis. 2009. “Blogs as Predictors of Movie Success.” In *Proceedings of the Third International AAAI Conference on Web and Social Media*. Association for the Advancement of Artificial Intelligence. http://ilpubs.stanford.edu:8090/906.

Sakai, Tetsuya. 2007. "On the reliability of information retrieval metrics based on graded relevance." *Information processing & management* 43, no. 2: 531-548.

Scaffidi, Christopher, Kevin Bierhoff, Eric Chang, Mikhael Felker, Herman Ng, and Chun Jin. 2007. “Red Opal: Product-Feature Scoring from Reviews.” In *Proceedings of the 8th ACM Conference on Electronic Commerce*, 182–91. ACM. http://dl.acm.org/citation.cfm?id=1250938.

Sirmakessis, Spiros. 2004. *Text Mining and Its Applications Results of the NEMIS Launch Conference*. Berlin, Heidelberg: Springer Berlin Heidelberg. http://dx.doi.org/10.1007/978-3-540-45219-5.

Socher, Richard. 2014. "RECURSIVE DEEP LEARNING FOR NATURAL LANGUAGE PROCESSING AND COMPUTER VISION." PhD, STANFORD UNIVERSITY.

Spina, Damiano, Edgar Meij, Maarten de Rijke, Andrei Oghina, Minh Thuong Bui, and Mathias Breuss. 2012. "Identifying Entity Aspects in Microblog Posts." In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1089–90. ACM. http://dl.acm.org/citation.cfm?id=2348483.

Tumasjan, Andranik, Timm Oliver Sprenger, Philipp G. Sandner, and Isabell M. Welpe. 2010. "Predicting Elections with Twitter: What 140 Characters Reveal about Political Sentiment." *ICWSM* 10: 178–85.

Turney, Peter. 2001. "Mining the Web for Synonyms: PMI-IR versus LSA on TOEFL." http://nparc.cisti-icist.nrc-cnrc.gc.ca/npsi/ctrl?action=rtdoc&an=5765594.

Turney, Peter D. 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*, 417–24. Association for Computational Linguistics. http://dl.acm.org/citation.cfm?id=1073153.

Wilson, Theresa, 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*, 347–54. Association for Computational Linguistics. http://dl.acm.org/citation.cfm?id=1220619.

Yano, Tae, and Noah A. Smith. 2010. "What's Worthy of Comment? Content and Comment Volume in Political Blogs." In *ICWSM*. http://www.aaai.org/ocs/index.php/ICWSM/ICWSM10/paper/download/1503/1898/.

Zaki, Mohammed J., and Wagner Meira. 2014. *Data Mining and Analysis: Fundamental Concepts and Algorithms*. New York, NY: Cambridge University Press.

Zhao, Wayne Xin, Jing Jiang, Jing He, Yang Song, Palakorn Achananuparp, Ee-Peng Lim, and Xiaoming Li. 2011. "Topical Keyphrase Extraction from Twitter." In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, 379–88. Association for Computational Linguistics. http://dl.acm.org/citation.cfm?id=2002521.

VITA AUCTORIS

Chukwuma Ejieh was born in 1990 in Ile Ife, Nigeria. He received his Bachelor's degree (Double Hons) in Computer Science with Mathematics from Obafemi Awolowo University, Ile Ife, Nigeria in 2012. He completed his M.Sc in Computer Science from the University of Windsor, Ontario Canada in May 2016. His research interests include Data Mining, Machine Learning, Information Retrieval and Natural Language Processing.