University of Cincinnati

Date: 6/1/2012

I, Srijan Nepal , hereby submit this original work as part of the requirements for the degree of Master of Science in Computer Science.

It is entitled:

Linguistic Approach to Information Extraction and Sentiment Analysis on Twitter

Student's name: **Srijan Nepal**

This work and its defense approved by:

Committee chair: Kenneth Berman, PhD

Committee member: Fred Annexstein, PhD

Committee member: Anca Ralescu, PhD

2721

Linguistic Approach to Information Extraction and Sentiment Analysis on Twitter

by

Srijan Nepal

A thesis submitted in partial satisfaction of the
requirements for the degree of
Master of Science

in

Computer Science and Engineering

in the

School of Computing Sciences and Informatics
of the
College of Engineering
of the
UNIVERSITY OF CINCINNATI, OHIO

Committee:
Dr. Kenneth Berman, Chair
Dr. Fred Annexstein
Dr. Anca Ralescu

Spring 2012

Abstract

Social media sites are one of the most popular destinations in todays online world. With millions of users visiting social networking sites like Facebook, YouTube, Twitter etc. every day to share social content at their disposal; from simple textual information about what they are doing at any moment of time, to opinions regarding products, people, events, movies to videos and music, these sites have become massive sources of user generated content. In this work we focus on one such social networking site - Twitter, for the task of information extraction and sentiment analysis.

This work presents a linguistic framework that first performs syntactic normalization of tweets on top of traditional data cleaning, extracts assertions from each tweet in the form of binary relations, and creates a contextualized knowledge base (KB). We then present a Language Model (LM) based classifier trained on a small set of manually tagged corpus, to perform sentence level sentiment analysis on the collected assertions to eventually create a KB that is backed by sentiment values. We use this approach to implement a contextualized sentiment based yes/no question answering system.

This work is dedicated to my parents, my lovely sister and my baba,

the people who mean the most to me.

Acknowledgments

First and foremost I would like to thank my advisor, Dr. Kenneth Berman for his continuous support during my study and research. With his enthusiasm, inspiration, ideas, immense knowledge and a wonderful sense of humor, he helped in making my work so much easier.

I would also like to thank my thesis committee members, Dr. Anca Ralescu and Dr. Fred Annexstein for their wonderful help, comments and feedback.

My sincere thanks also goes to Dr. George Stan and Dr. Ruxandra Dima, from the Department of Chemistry, for providing me an opportunity to work for them in their research labs that supported me during those broke college student days.

My thanks also goes to my labmates, Akash Kushawaha and Jasman Kaur for their constant encouragement and my very good friends Madhan and Sanshit, who've been with me through a lot of thick and thins.

And a special thanks goes to all my friends from 452 Riddle Road and the small Nepali community around Clifton who helped me stay sane during those crazy days. You guys definitely deserve a very loud thank you.

And I would finally like to thank my parents who've sacrificed a lot to see me reach this day. To them, I dedicate this work.

Contents

List of Figures **vii**

List of Tables **viii**

1 Introduction **1**
1.1 Motivation 1
1.2 Objective 4
1.3 Thesis Structure 5

2 Background **6**
2.1 Twitter 6
2.1.1 Twitter Terms and Concepts 7
2.2 State-of-the-Art 9
2.2.1 Open Information Extraction 9
2.2.2 Information Extraction from Twitter 10
2.2.3 Classical Sentiment Analysis 11
2.2.4 Twitter Specific Sentiment Analysis 12
2.2.5 Online Tools for Twitter Sentiment Analysis 16
2.3 Assumptions 20

3 3 Implementation **21**
3.1 Overview 21
3.2 System Architecture 22
3.3 Technology 23
3.4 Data Collection 25
3.5 Data Cleaning and Feature Reduction 26
3.5.1 Language Identification 26
3.5.2 POS Tagging 26
3.5.3 Regular Expression 27
3.5.4 Machine Translation of Tweets 28

3.6 Information Extraction 29
3.6.1 Syntactic Parser based Relation Extraction 29
3.6.2 ReVerb based Information Extraction 30
3.7 Sentiment Analysis 33
3.7.1 Naive Bayes Classifier 33
3.7.2 Support Vector Machine (SVM) 34
3.7.3 Language Model based Classifiers 35
3.8 Question Answering 36

4 4 Experiments and Analysis 38
4.1 Sentiment Analysis 38
4.1.1 Sentiment Corpus 38
4.1.2 Corpus Analysis 39
4.2 Sentiment Cloud 41
4.3 Experimental Design 41
4.3.1 Naive Bayes Classifier 42
4.3.2 Support Vector Machine (SVM) 43
4.3.3 Character N-gram LM Classifier 43
4.3.4 Token Language Model Classifier 44
4.4 Comparison 44
4.5 Question Answering 44

5 Conclusion and Future Work 50
5.1 Limitations 51
5.2 Future Work 52

Bibliography 53

List of Figures

2.1 A snapshot of Twitter 7
2.2 Sentiment140 17
2.3 Tweetfeel 18
2.4 Twitrratr 19

3.1 System Architecture 23

4.1 Word Frequency 40
4.2 Will Obama win in Ohio? (Week of April 15-21, 2012) 47
4.3 Positive Sentiment Word Cloud 48
4.4 Negative Sentiment Word Cloud 49

List of Tables

3.1 Sample Data Cleaning Results 28
3.2 Sample Machine Translation Results 29
3.3 Triplet Extraction Algorithm 30
3.4 Sample Binary Relation Extraction using OpenNLP Parser 31
3.5 Reverb Relation Extractor for Twitter [16] 32
3.6 Sample ReVerb Extractions 32

4.1 Performance of Naive Bayes Sentiment Classifier 42
4.2 Performance of SVM 43
4.3 Performance of Character N-gram Language Model 44
4.4 Token N-gram Language Model based Classification 44
4.5 Performance Comparison with some previous works 45
4.6 Will Obama win in Ohio? (Week of April 15-21, 2012) 46

Chapter 1

Introduction

1.1 Motivation

Social media sites are one of the most popular destinations in todays online world. Millions of users visit social networking sites like Facebook, YouTube, Twitter etc. every day to share social content at their disposal - from simple textual information about what they are doing at any moment of time, to opinions regarding products, people, events, movies and so on. This widespread use of social media applications by people all across the globe has caused these sites to become massive sources of user generated content. In this thesis we focus on one such social networking site, Twitter[1]. We study information posted on Twitter to check if it can act as a reliable source for creating a contextualized knowledge base. We specifically study

[1]http://www.twitter.com

information posted on Twitter regarding the 2012 United States presidential election and track the news and opinions around the involved candidates and political parties.

Twitter is a micro-blogging site that has become phenomenally popular over the last few years. At the heart of Twitter is a service called tweeting, which basically means posting a short textual message called tweet to a group of followers. Since its inception in 2006, the site has grown at an unprecedented rate. Today people post the 140-character long messages or tweets more than 200 million times a day [5]. With millions of people tweeting their reaction on perceived happenings, situations, current scenarios, products, expressing their political and religious beliefs and opinions, Twitter has become one massive corpus of current information and opinion. While the accepted fact is that majority of tweets are pointless babbles, are conversational in nature or are spams, it has been found that approximately 3.6% of them are on topics of mainstream news and events [4]. Thanks to the proliferation of mobile devices and overall ease of access to the web, barrier to tweeting has become extremely low and people all across the globe are using Twitter to post information at their disposal for the consumption of the global audience; the Arab Spring of 2011 being a point in case. Today Twitter has become the fastest source of news for majority of people with any sort of regular online habit [22].

Tapping this huge wealth of knowledge promises significant opportunities and in the

same time presents significant challenges as well. Tweets are short, contain informal grammar and slangs and involve a language that is constantly evolving. Analyzing and extracting important information from such a huge and noisy corpus is a massive task. Although a lot of progress has been made in the field of Natural Language Processing (NLP) and Information Extraction (IE), state-of-the-art NLP and IE tools tend to perform poorly when dealing with out-of-domain and noisy data like tweets [36]. With this in mind, in this thesis we present a framework that allows for important information extraction from tweets. We extract assertion(s) or fact(s) from individual tweet in the form of binary relations. Using existing NLP and IE techniques we identify relation phrases, phrases that represent relations between entities in English language, and the binary entities that particular relation connects. This triplet,

(argument1-relation phrase-argument2),

is the building block of the contextualized KB we aim to create using Twitter, with each relation representing a singular fact extracted from tweets. We then perform sentiment analysis on the collected relations and categorize them under one of the following classes - positive, neutral and negative. Almost all prior sentiment analysis work done on Twitter are based on identifying subjective tweets. The idea is to first identify if a given Tweet is opinionated or not and perform sentiment analysis only if it is determined that the tweet does indeed contain opinion. We perform sentiment analysis on the extracted relation irrespective of the subjectivity of it. That is, given

any fact or news, our aim is to find out if it is positive, neutral or negative towards the entity mentioned in it. The fact may be an opinion or a general piece of news. Specifically the interest is in determining if any given fact, extracted from news or opinion, is positive, negative or neutral towards a particular US presidential candidate mentioned in that particular fact. As an example, consider the following tweet.

> @ukobserver: The independence base is already Xs but Y has lost his mind and Americans.

Here, instead of assigning a single sentiment label to the entire tweet, we first identify relations (argument1-relation phrase-argument2) embedded in it and then perform sentiment analysis on the extracted relations. From the above tweet, the relations extracted would be:

> The Independence base - is already - X's
> Y - has lost - his mind
> Y - has lost - Americans

And our approach, unlike most other Twitter based sentiment analysis approaches, classifies the individual relations rather than the whole tweet itself. So each of the above relations will be classified separately.

1.2 Objective

In this work we present a Twitter specific Information Extraction (IE) and Natural Language Processing (NLP) framework using which we extract important facts from

a collection of tweets. We then create and evaluate supervised classifiers that are able to assign sentiment values to those facts. Finally we demonstrate that using the knowledge base created from facts extracted from Twitter, contextualized sentiment-based yes-no questions as well as other general knowledge questions can be answered. The overall objective of this thesis is to demonstrate a new granular sentiment analysis approach for Twitter.

For practical purpose, we collected tweets around the 2012 US presidential election from January 2012 to May 2012. We then extracted assertions and performed sentiment analysis on those extractions to create a KB. And using this very KB, we demonstrate a system that can answer sentiment based yes no questions as well as general factual questions limited to a particular context, which in our case is the 2012 US presidential election.

1.3 Thesis Structure

This thesis is organized as follows. There are a total of 5 chapters. In chapter 2, we introduce Twitter and concepts and terms that are associated with it and we discuss related work and the theoretical foundations of this thesis. Chapter 3 presents our methodology and discusses our complete implementation process. In chapter 4, we describe our experiments and results and analysis of those results. Finally in chapter 5 we provide future work and conclusion.

Chapter 2

Background

2.1 Twitter

Microblogging is one of the most popular forms of online social communication in today's world in which users normally post short textual messages of up to a few hundred characters mostly updating their circle of friends or followers with updates about their daily life, happenings around them and reaction to news and events they might have witnessed. Twitter is undoubtedly the most popular microblogging site today [22]. It essentially is an online social networking and microblogging service that allows its users to post short textual messages, 140 characters long, called tweets. With a user-base of around 300 million and generating over 200 million tweets per day [5], Twitter has become one of the most powerful social networking services. Figure 2.1 shows Twitter's front page.

Figure 2.1: A snapshot of Twitter

2.1.1 Twitter Terms and Concepts

Tweet

Twitter allows its users to post short text messages called tweets. These messages can be at most 140 characters long. Using this short text, users post summary of what they are doing or what they are observing at the given point of time. Because of the size limit, tweets normally contain abbreviations, slangs, shortened URLs that point to documents that may contain more detailed information, textual emoticons to demonstrate the user's sentiments and so on. A user may label his tweet as private,

in which case his messages appears to only his followers and if the user chooses to make his tweets public, they can be viewed by any other user.

Followers

Twitter employs a model called follower to create its social network. The follower model is different from friendship model that most other social networks employ in that a Twitter user can follow any other user without having that person follow him back. When you as a Twitter user follow another user, the status updates of that person start appearing in your Twitter feed.

Hashtags

In Twitter, users can tag their messages using hashtags. Hashtags are words or phrases that begin with a hash (#) sign. For example, consider the following tweet.

"#Reds closer #RyanMadson out for the season".

This tweet has been tagged with words Reds and RyanMadson. Now anyone searching for Reds or RyanMadson can potentially view this tweet.

Twitter Search API

Twitter produces over 200 million tweets a day [5]. And to allow developers access to this huge corpus of data, Twitter also provides a well-defined set of search APIs. The regular search APIs allow developers to search for tweets that mention some

particular keywords, are posted by some particular users or are referencing certain users. Similarly, Twitter also provides Streaming APIs. These APIs allow users to get real time sample from the Twitter Firehose [47]. The streaming APIs allow for large quantities of keywords to be specified while searching for twitter and also allows for searching geo-tagged tweets.

Twitter Information Content

It has been found that the vast majorities of tweets are simply pointless babbles or are conversational in nature (78%) and only 4% are concerned with news and current events [4]. And like any other social networking services, Twitter has its fair share of problems with spams. They mostly include aggressive following, self promotion, links to phishing and malware sites and unsolicited and unwanted advertisements.

2.2 State-of-the-Art

2.2.1 Open Information Extraction

Recently a lot of work has been done in the field of Open Information Extraction (IE) with most research focusing on web documents. Traditional Information Extraction systems normally take a relation name and hand-tagged examples of that relation as input [8]. This mechanism is focused on satisfying a precise, narrow, pre-specified request from small corpora and is domain dependent; like finding location and time

of seminars [7, 26]. On the other hand Open IE is not dependent on any relation and can extract relations from massive and heterogeneous corpora without any relation-specific input [8, 7]. TextRunner [7] and WOE [45] are examples of early successful Open IE systems that demonstrated the capability of binary relations extraction from general web documents and Wikipedia, respectively. Two types of errors, incoherent and uninformative extractions were the major issues with these early systems [16]. Fader, Soderland and Etzioni [16] then introduced ReVerb Extractor, another Open IE system that was able to improve on its predecessors work by introducing syntactic and lexical constraint in the process of relation extraction. By following a simple approach of extracting relation first and then corresponding arguments connected by it and employing easy-to-enforce constraints on binary, verb-based relation phrases in English, ReVerb was able to outperform previous Open IE systems in both recall and precision [16]. Exploiting their findings that the majority of arguments fit into a small number of syntactic categories and using common delimiters to identify argument boundaries, the authors [16] were able to design and add argument learning component to ReVerb [15].

2.2.2 Information Extraction from Twitter

Named Entity Recognition (NER) and Part-Of-Speech (POS) tagging in Twitter are active research fields today [42, 19]. Similarly quite a few efforts have been made in summarizing Twitter topics [44, 13]. Ritter, Clark, Mausam and Etzioni have shown

that existing tools for POS tagging, Chunking and NER perform quite poorly when applied to Tweets. The authors have come up with a system built on annotated Tweets, trained on unlabeled, in-domain and out-of-domain data showing substantial improvement over news-corpus trained traditional tools [19].

Gimpel et al [42] have developed a POS tagger for Twitter by manually tagging 1827 tweets. The authors [42] came up with their own annotation scheme and coarse tags targeted especially towards Twitter that captures Twitter-specific properties.

Sharifi, Hutton and Kalita [44] have built an algorithm that summarizes a collection of tweets around a particular context, represented by a keyword or phrase by performing word and phrase level analysis. Similarly, Chakrabarti and Punera [13] have shown that it is possible to construct real-time summaries of events from Twitter posts based on learning the underlying hidden state representation of an event in Twitter.

2.2.3 Classical Sentiment Analysis

A lot of prior work can be found in the field of sentiment analysis. Most of the earlier works are concerned with more generic, document level sentiment analysis, where the task is to classify a given document into one of available sentiment categories [37, 34, 35]. Other sentiment analysis research involve, word level sentiment analysis, classifying subjective and objective sentences, general sentence level sentiment analysis or clause level sentiment analysis [36]. The latter tasks analyze sentiments at a more granular level and have shown much more effective classification capabilities

[36]. Data used by most of these works were from movie and product reviews or more formal corpus like the Wall Street Journal corpus, Multi-Perspective Question Answering (MPQA) corpus and Document Understanding Conference (DUC) corpus [39].

2.2.4 Twitter Specific Sentiment Analysis

Thanks to the speed and volume at which opinions and information regarding a product, an event, a movie, a person and so on can be collected using Twitter, it is no surprise that there have been a flurry of recent works on Sentiment Analysis on Twitter. Pak and Paroubek [33] used multinomial Naive Bayes classifier to classify tweets into positive, negative and neutral categories. They used N-gram and POS-tags as feature set for their experiments. However, Kouloumpis, Wilson and Moore, in their aptly named paper [28] have shown that POS features may not be useful for sentiment analysis in the microblogging domain. Instead they suggest using existing sentiment lexicon with microblogging features like emoticons and abbreviations. Barbosa and Feng [9] used two phased approach to sentiment analysis using Twitter. They first classified tweets into objective and subjective classes and then performed sentiment analysis on subjective dataset only using several classifiers including SVM and found that Naive Bayes gave the best result. Go, Bhayani and Huang [20], used emoticons as noisy labels for training classifiers based on Naive Bayes, maximum entropy and SVM.

Preprocessing

All the steps performed after obtaining the Tweets using Twitters search APIs are labeled as pre-processing. For traditional data cleaning operations, POS tagger and regular expression are used. We follow the same process and use regular expressions and POS tagging for Twitter [42] to strip each tweet of all data and characters that we consider as noise.

Regular Expression

Regular expressions are context-independent syntax that can represent a wide variety of character sets and character set orderings [21]. They basically provide a mechanism to select specific strings from a set of character strings [21]. Based on the nature of substring that needs to be extracted, a pattern is created that adheres to the rule of regular expression and using that pattern the target text is processed. Regular expression can be used to search for a pattern in a text, do a modification or extract matching texts.

Features

In sentiment classification or for that matter any text processing it is imperative that a piece of text is converted into a feature vector that exhibits the most important features available [36]. This basically reduces the training duration and also removes noise by getting rid of irrelevant features. There exists a vast body of work that

discusses feature engineering in detail [18]. In this section we discuss those features that are of importance in our context.

Part-of-Speech(POS)

Part-of-speech (POS) tagging is one of the most fundamental components of linguistic analysis pipeline. It is a simple form of syntactic analysis in which a word in a given text is marked by its part of speech in the concerned language such as noun, verb, adjective etc., based on both its definition as well as its context [46]. POS information is commonly exploited in sentiment analysis and opinion mining [36]. A high correlation has been found between subjectivity of a sentence and presence of adjectives and it has also been observed that adjectives are good indicators of sentiment [23]. Similarly comparisons of the effectiveness of adjectives, verbs and adverbs in sentiment detection has also been studied [10, 32, 49].

TF-IDF

In a given body of text, a document or even a single sentence, each term has a different weight, based on the value it adds to the overall document. Term Frequency -Inverse Dcocument Frequency (TF-IDF) is used to measure such importance. Essentially, TF-IDF works by determining the relative frequency of words in a specific document compared to the inverse proportion of that word over the entire document corpus. Basically this calculation determines how relevant a given word is in a particular

document. Words that are common in a single or a small group of documents tend to have higher TF-IDF numbers than common words such as articles and prepositions [41].

N-grams

An n-gram is a co-occurring sequence of n items from a given sequence of text. The items can be almost anything but in most cases, especially in text categorization, character n-grams and word n-grams are preferred. Character n-grams are n adjacent characters in a given input string. Similarly, word or token n-grams are n adjacent words or tokens in a given body of text. N-grams are particularly useful for sentiment classification where you can determine the sentiment value carried by individual n-gram. In our sentiment classification experiments we used presence of n-grams as features. Pang et al. [37] have shown that in text classifiation n-gram presence as feature outperform n-gram frequency.

Sentiment Classification Methods

Sentiment classification is done either by supervised or unsupervised machine learning techniques. In unsupervised technique, a function which compares the features of a given text against discriminatory-word lexicons whose polarity are known beforehand is used.

In supervised technique, you build a classifier. The classifier needs training examples

which can be labeled manually or obtained from user generated labeled source. Pang et. al [34] have shown that supervised techniques tend to outperform unsupervised techniques. For our study we will be using supervised machine learning methods. We will demonstrate the use of Token N-gram based Language Model, Character N-gram based Language Model, Naive Bayes classifier and Support Vector Machines (SVM) and show which method performs better.

2.2.5 Online Tools for Twitter Sentiment Analysis

Sentiment140 Based on Go, Bhayani and Huang's work [20], Sentiment140 uses machine learning approach to sentiment classification. By creating a sentiment corpus based on positive and negative emoticons, the authors were able to achive accuracy of over 80% by using different machine learning algorithms - Naive Bayes, Maximum Entropy and SVM.

Figure 2.2: Sentiment140

Tweetfeel Tweetfeel is another sentiment analysis tool for Twitter. It uses its own proprietary algorithm to find the sentiment of any given tweet. Figure 2.3 shows Tweetfeel's sentiment results for keyword "Obama".

Figure 2.3: Tweetfeel

Twitrratr Twitrratr also performs sentiment analysis on Tweets but rather than just classifying tweets as being positive or negative, it also groups them under neutral label; that is, it performs multi-level sentiment classification of tweets. Figure 2.4 shows Twitrratr's grouping of tweets under different sentiment classes for keyword "Romney".

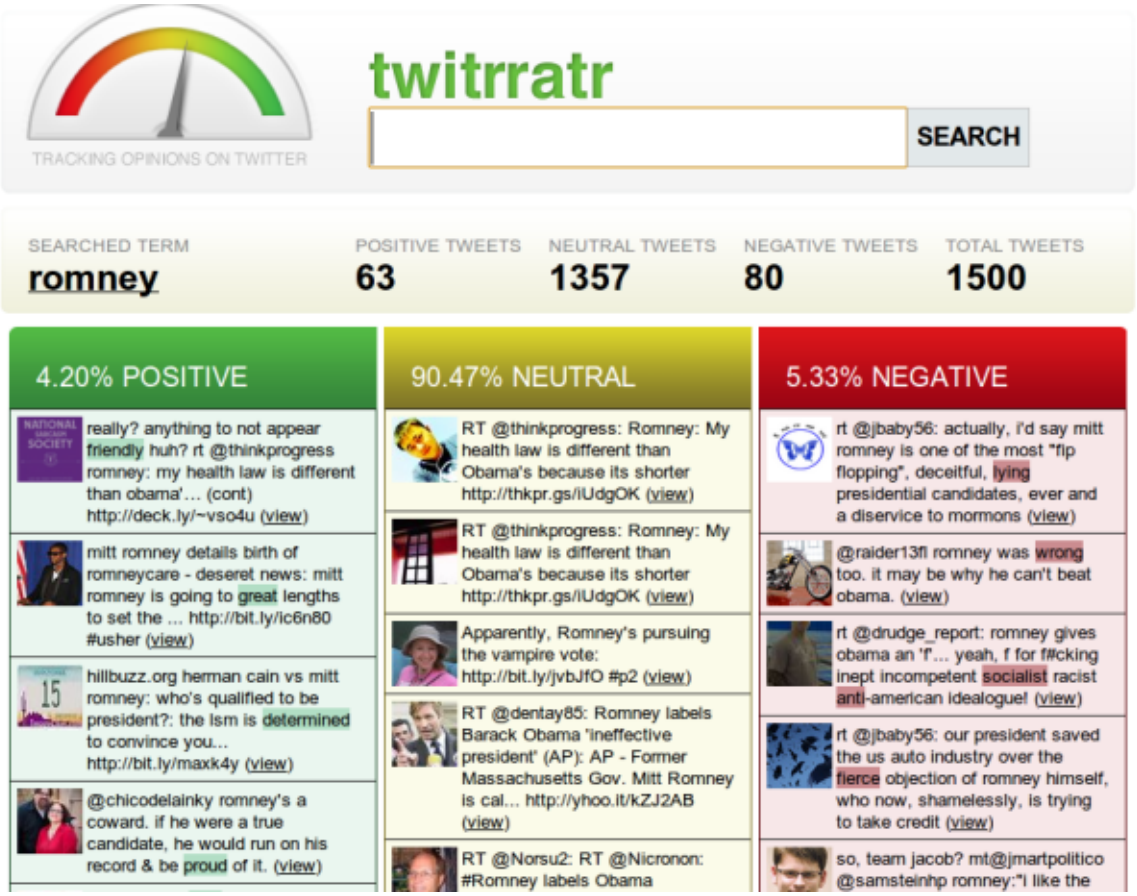

Figure 2.4: Twitrratr

2.3 Assumptions

For the purpose of our study, we have had to make certain assumptions. Our approach to sentiment classification is slightly different from traditional approaches. Instead of labeling a tweet as being positive, negative or neutral, irrespective of the context, we try to assign sentiment label to the tweets based on the enitity mentioned in it. For that purpose, we do not discriminate between subjective and objective assertions. Most prior sentiment classification experiments first classify a given text into subjective or objective classes and perform sentiment analysis only if the text is found to be subjective. It works well for opinions, but if your aim to identify if a piece of news or a fact is positive, negative or neutral towards a particular entity, assigning sentiment value to objective texts is very important. That is what we do in this experiment. Taking the US presidential election as our context, we treat each assertion as positive, neutral or negative towards the candidate or party mentioned in that assertion. And we also assume that, each assertion we extract, will concern a single individual or an entity.

Chapter 3

3 Implementation

3.1 Overview

The implementation section of this thesis discusses the architecture of our framework. Using the framework described in this chapter, we collect tweets around a given topic or a list of topics, extract facts or assertions in the form of binary relation triplet, and run these relations and not the tweets, through several supervised classifiers to categorize them into positive, negative or neutral classes and create a contextualized Knowledge Base (KB). A sample experimental module that exhibits the capability of this framework to answer contextualized questions is also included as part of the framework.

3.2 System Architecture

Using the publicly available Java wrapper for Twitter Application Programming Interface(API), Twitter4J[2], the system first collects tweets from the Twitter Firehose, the complete stream of all the tweets coming out from Twitter [48], using the streaming API [47]. The collected data is subjected to cleaning and feature reduction process where we remove those Twitter features that we regard as noise for our information extraction and sentiment analysis sub-processes. The cleaned tweets are then syntactically normalized using a n-gram language model based machine translation system, Moses[3], to convert known internet slangs, lingos and abbreviations to proper English format. We then subject the normalized tweets to our Information Extraction(IE) module. Using two separate linguistic approaches - first based on syntactic parsing and the second based on an open source multi-corpora IE system, ReVerb[4], we extract assertions from the collected tweets in the form of binary relations. Then using several supervised learning algorithms and sentence level sentiment analysis methods, the extracted relations are classified into one of positive, negative or neutral sentiment categories and stored into a database to form our contextualized knowledge base.

[2]http://twitter4j.org
[3]http://www.statmt.org/moses
[4]http://reverb.cs.washington.edu

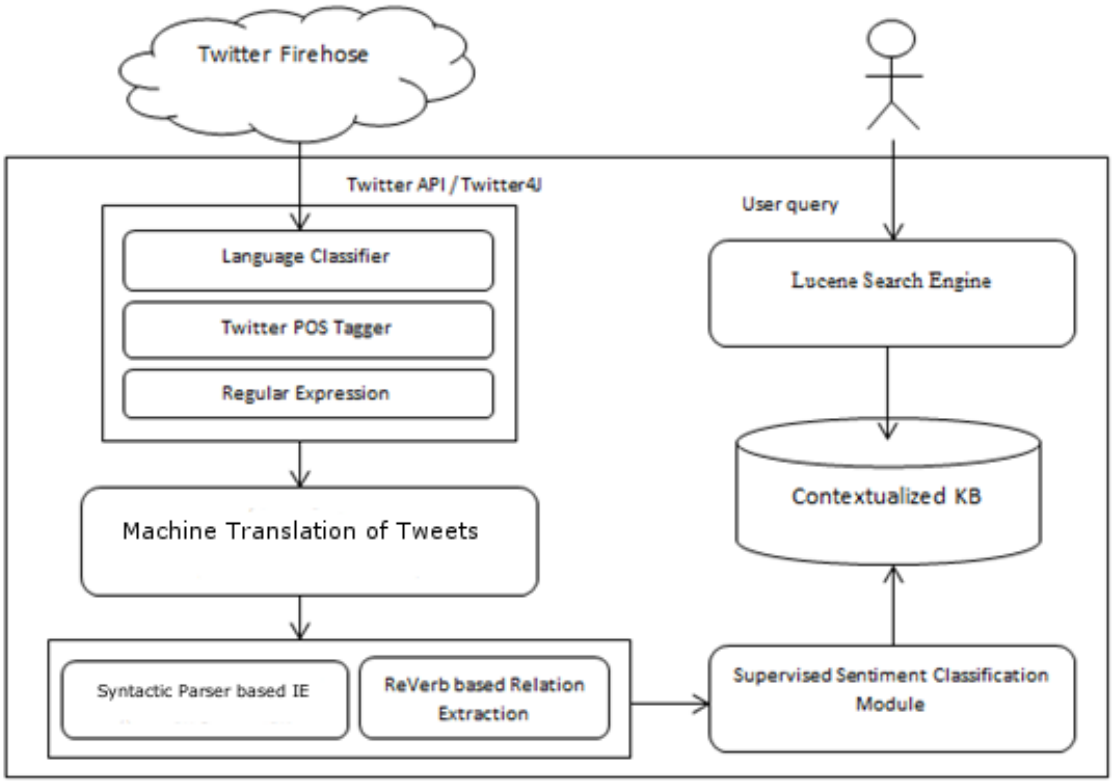

Figure 3.1: System Architecture

3.3 Technology

Java

Java is one of the most popular programming languages today. We selected Java to implement our framework so that we would be able to exploit the huge cache of publicly available Java libraries for Natural Language Processing, Information Extraction, text processing and mining, tweet collection, database operations and other general purpose libraries.

R

R is an open source statistical programming language. It provides support for linear and non-linear modeling, classification, clustering and other statistical analysis operations. In our case, R was very helpful in general purpose text mining operations and for corpus creation. For further details regarding the R language, see [14] and [2].

Twitter4J

Twiter4J is an open source unofficial Java library for Twitter Application Programming Interface(API). It provides Java wrapper functions for almost all Twitter APIs and thus allows for seamless integration of Twitter API into any Java based applications. Twitter provides APIs that support both REST and streaming services. The streaming API [6] provides real-time filtered sample from the Twitter firehose. The API is HTTP based and GET, POST, DELETE requests can be used to access the tweets [11]. Twitter4J supports both the REST and the streaming APIs

OpenNLP

The Apache OpenNLP[5] library is a Java based machine learning toolkit that provides natural language processing capabilities. It supports normal NLP tasks like tokenization, sentence segmentation, POS tagging, Named Entity Recognition (NER), chunking, parsing and co-reference resolution. Further information about OpenNLP

[5]htp://opennlp.apache.org

can be found in [3].

Moses

Moses is a statistical machine translation system that allows users to train a translation system for any language pair. It requires a parallel corpus using which one can train a translator. Using N-gram language model, it then translates one language into another by finding phrases in target language that correspond to phrases in the source language [27].

ReVerb

ReVerb is an Open Source Java application that can identify and extract binary relationships from English sentences. Unlike other Open IE systems that are trained on a limited set of relations and their corresponding arguments, ReVerb exploits its knowledge of relation specifying phrases, that is, words in English language that represent relations. It then traverses the left and right side of relation in a sentence to identify arguments that are connected together by the relation [16, 15].

3.4 Data Collection

To collect tweets around the US presidential election 2010, we made use of Twitter4J library. We collected 2000 tweets every day from late January to May, 2012. The tweets, along with the users Twitter name, timestamp and geo-location (when

available) were saved in simple CSV files.

3.5 Data Cleaning and Feature Reduction

Tweets generally contain a plethora of informal Internet lingos that do not make sense to traditional natural language processing and information extraction systems that have been trained on formal language based corpus. But tweets have certain structure and unique features that can be exploited to get rid of a lot of contents that do not add value to their information content. For example in most cases, username or hashtags can be safely removed as they dont tend to add to the informational content of the tweets. To obtain meaningful extractions from tweets, we therefore subjected the collected tweets to a series of data cleaning steps.

3.5.1 Language Identification

First thing we did in the process of data cleaning was selecting only English tweets. Using Lingpipes [1] character language model based text classifier and Leipzig Corpus[7], we selected only those tweets that were most likely to be English.

3.5.2 POS Tagging

The second phase of data cleaning involved POS tagging of each individual tweet. Certain features of tweets like "@username" and "#hashtags" can be safely removed

[7]http://corpora.uni-leipzig.de

from tweets as in most cases they do not add to the information content. But unfortunately that is not always the case. For instance, lets look at the following examples.

> Tell @MittRomney bring your offshore account back to USA #Obama
> Too late for GOP to undermine #Obamas youth connection so they will try to undermine youth turnout instead.

In the first example, @MittRomney is not something that you would want to remove. The word has a valid syntactic role in the sentence. But in the same tweet, the hashtag #Obama has no syntactic role at all. But in the second example, hashtag #Obamas has a valid syntactic role. To solve this problem of determining what words are meaningful and what words are not, we used a Twitter based POS tagger [19]. After tagging individual tokens of a tweet with different POS tags, we were able to remove hashtags, usernames, emoticons, URLs that do not have any syntactic meaning in the individual tweets.

3.5.3 Regular Expression

Twitter users tend to use word repetition to convey their feeling in a much stronger way. To a human observer, the repetitions may add to the strength of words being used, but for a text processing system, these repetitions do not make any sense. In the second phase of data cleaning, we removed all words from the collected tweets that had a single word repeat consecutively at least four times. For example, words like "soooo", "looool", "OMGGGGG" are removed using regular expression. Similarly we observed that users tend to merge a lot of words, especially hashtags together to

form one large word or hashtag. In most cases these words did not seem to make sense. So we decided to get rid of any word that was longer that 15 characters in length. And eventually all non-ascii characters, if any, were removed. See table 3.1 for examples of the data cleaning process.

Original Tweet	**Cleaned Tweet**
RT @mmfa: Sen majority voted in favor of Buffett rule. GOP filibustering ensured rule's defeat. Media's response: "Meh" http://t.co/Kt21WIQA	sen majority voted in favor of buffett rule . gop filibustering ensured rule's defeat . media's response meh
The #Romney endorsement by Coburn and Cantor confirms that the RINO GOP elites want to wrap up the nomination for Romney	the romney endorsement by coburn and cantor confirms that the rino gop elites want to wrap up the nomination for romney
RT @Saint_Obama: We Need Newt Now! <<<<http://t.co/WSw5gVQ3 via @youtube ¡¡¡¡ NEWTWEST 2012 #Women4Newt #CT #D ...	we need newt now ! via newtwest 2012

Table 3.1: Sample Data Cleaning Results

3.5.4 Machine Translation of Tweets

The output obtained from data cleaning operation is still not good enough for traditional natural language processing or information extraction systems as they still have informal words, slangs, abbreviations and other informal characters and tokens. Using Moses, a statistical machine translation system, and a manually created SMS to English parallel corpus, kindly provided by [40], we translated each Tweet to its

English equivalent. Moses works with a corpus in the target language, using which it builds a n-gram language model. It then requires a set of parallel corpora, one in the source and one in the target language. Using the SMS-English corpora created by [40], and English corpus from American National Corpus [24] and based on the work done by Kaufman [27], we were able translate the collected tweets into standard English like representations. See table 3.2 for examples of machine translation of tweets into English.

Cleaned Tweet	**Translated Tweets**
u r amazing 2	you are amazing too
m voting for Obama 2morrow	I'm voting for Obama tomorrow
I'm comin for ur job	I'm coming for your job
I aint voting Romeny	I am not voting Romney
They aint comin 2day	They are not coming today

Table 3.2: Sample Machine Translation Results

3.6 Information Extraction

The next phase involves assertion extraction from the cleaned tweets. By using traditional NLP method and modern Open IE system, we demonstrate how meaningful information can be extracted from tweets.

3.6.1 Syntactic Parser based Relation Extraction

We first applied modified version of Triplet Extraction algorithm developed by [43]

Algorithm for extracting triplets from tweets

1. For each cleaned tweet t, obtain a Treebank based parse tree, P
2. Now for each individual simple declarative clause S extracted from P, having NP and VP sub-trees
3. Do Extract
    - Relation : Deepest verb from the VP sub-tree and its attributes
    - Subject: First noun in the NP sub-tree and its attributes
    - Object:Find noun or adjective elements in the sibling of the VP sub-tree and its attributes.
    - Return relation form triplet (Subject,Relation,Object) or failure
4. Done

Table 3.3: Triplet Extraction Algorithm

Table 3.4 demonstrates example outputs of the triplet extraction algorithm shown in 3.3.

3.6.2 ReVerb based Information Extraction

Our another relation extraction approach involved, ReVerb Open IE system that identifies relations based on its prior knowledge of relations that are used in English language. ReVerb first identifies relation in a statement and then goes about finding the most probable binary arguments that are linked by the extracted relation. Table 3.5 shows the working of the modified version of ReVerb algorithm for Twitter.

To identify and extract relations, ReVerb follows a simple syntactic constraint rule. A relation r has to be either a verb, a verb followed by preposition, or a verb followed by nouns and adjectives or adverbs. Similarly to stop over-simplification of relation

Sentence	**Parse Tree**	**Extraction**
Obama is in Ohio	(S(NP(NNP Obama))(VP(VBZ is)(PP(IN in)(NP(NNP Ohio))))))	obama, is in, ohio
Prez Barack Obama is in Ohio	(S(NP(NNP Prez)(NNP Barack)(NNP Obama))(VP(VBZ is)(PP(IN in)(NP(NNP Ohio))))))	prez barack obama, is in, ohio
Prez Barack Obama is in Ohio today	(S(NP(NNP Obama))(VP(VBZ is)(PP(IN in)(NP(NNP Ohio)(NN today))))))	prez barack obama, is in, ohio today

Table 3.4: Sample Binary Relation Extraction using OpenNLP Parser

phrases, it introduces a lexical constraint. This constraint forces the extracted relation to take only k distinct arguments. ReVerb learns this k from a large dictionary of relations in English language.

A logistic regression classifier is then used to assign confidence values to extracted relations. This classifier was separately trained on manually labeled extractions from 500 tweets. Each correct and incorrect extractions were assigned 1 and 0 respectively. So as to obtain higher number of extractions from tweets, we accept all extractions with a confidence value greater than 60%.

Table 3.6 shows example of relations extracted from cleaned and machine translated tweets.

Algorithm: ReVerb Open IE

1. Given a cleaned, syntactically normalized tweet T,
2. Do:Relation Extraction
    - For each verb v in T, find the longest sequence of words r, such that r starts at v and satisfies the syntactic and lexical constraints defined below
    - If any pair of matches are adjacent or overlap, merge them into a single match
3. Done
4. Do: Argument Extraction
    - For each relation phrase identified above, find the nearest noun phrase x to the left of r in T such that that x is not a relative pronoun, who-adverb or existential "there"
    - Then Find the nearest noun phrase y to the right of r in T.
    - if such pair (x,y) is found return (x,r,y)
    - if y not found, return x,r
    - if x not found, return r,y
5. Done

Table 3.5: Reverb Relation Extractor for Twitter [16]

Tweet	**Relations**
Obama may be the food stamp president, but newt wil be the commander in chief	- obama <may be >the food stamp president - newt <will be >the commander in chief
President Obama will win the november election	- president obama <will win >the november election
If you do not support mitt romney, you support obama. if you support obama, you are not conservative.	- you <do not support >mitt romney - you <support >obama
romney talking about soul? that would be like gingrich telling his opponents not to go negative	- romney talking about soul <would be like >gingrich telling his opponents not to go negative
romney puts foot in mouth, again	- romney <puts foot in >mouth

Table 3.6: Sample ReVerb Extractions

3.7 Sentiment Analysis

Now each such extracted assertion has to be assigned a sentiment label. For this purpose we created a text classifier using four different algorithms.

3.7.1 Naive Bayes Classifier

Naive Bayes method of classification is a very popular text classification method. It is a simple probabilistic classifier based on Bayes theorem with strong independence assumptions. That is, it assumes that presence or absence of a certain feature in a category is not related to the presence or absence of any other feature.

Naive Bayes Probabilistic Model

According to Bayes theorem,

$$posterior\ probability = \frac{prior\ probability \times likelihood}{evidence}$$

This can be written as,

$$p(C|F_1, ...F_n) = \frac{p(C)p(F_1,....,F_n|C)}{p(F_1,....,F_n}$$

where C is a dependent class variable with a small number of outcomes and $F_1, F_2....F_n$ are features.

Now, we can safely ignore the evidence part, as it does not depend on C. And by using the definition of conditional probability, the above relation can then be rewritten as,

$$p(C|F_1, ...F_n) = p(C)p(F_1|C)p(F_2|C, F_1)p(F_3|C, F_1, F_2)...p(F_n|C, F_1, F_2..., F_{n-1})$$

And as Naive Bayes model assumes each feature is conditionally independent of every other feature, the above relation can be written as

$$p(C) \prod_{i=1}^{n} p(F_i|C)$$

Finally for a given set of finite categories, the probability of a document d being in class C is computated by

$$arg\ max_c \prod_{i=1}^{n} P(F_i = f_i|C = c)$$

Here prior probability represents the fraction of appearances of that feature in different categories in the training set. For our experiments purpose we used the Naive Bayes Classifier from Java Mallet project[31]. We used unigram prsence, bigram presence and combination of both unigram and bigram presence as feature for Naive Bayes classifier.

3.7.2 Support Vector Machine (SVM)

SVM is a popular supervised machine learning technique that takes a set of data as input and predicts for each given input which two possible categories the input most likely belongs to. It basically creates a hyperplace to separate feature vectors of one class with another class, with the highest possible margin. The items which determine the margin are the support vectors and functions that map the given set of inputs into higher dimensional spaces such that a hyperplane which can separete the given data into binary classes can be found are the kernels. For further reading regarding

SVM, [12, 17] can be consulted. To implement SVM, We used the SVM_light package [25] with default settings.

3.7.3 Language Model based Classifiers

A language model(LM) assigns a probability to a sequence of characters or tokens. An n-gram model approximates these probabilities by assuming that the only characters or tokens needed to predict $P(c_i|c_1.....c_{i-1})$ are the first i-1 elements. Language model assigns probabilities to elements found in a given language. These elements can be bytes, sequences of characters or sequences of tokens or words. We used n-gram based character language model and token language model classifiers to classify extracted relations into three sentiment categories.

To perform text classification using language model, the basic idea is to create a language model for each category separately. When an unknown body of text needs to be classified, first a language model is calculated and compared against the LMs generated frmom the training set. The text is assigned to the category that most closely matches the existing LMs.

N-gram character language model based classification is a new method of text classification which is derived from N-gram language models. Instead of taking words as the basic unit, characters are chosen as the unit of operation. The N-gram character language model provides a probabilistic distribution defined for a string over a fixed set of alphabets. It basically stores counts for sequence of character of predefined size

in a given corpus. These counts are then used to predict the probability of a new sequence of characters. This simple measure is resistant to different types of textual errors like, spelling mistakes, different word order or valid mis-spellings, which makes it an ideal candidate to classify data extracted from noisy medium like Twitter.

Token n-gram language model based classification uses the n-gram count for sequence of tokens or words to classify given body of texts. From the training corpus, the model learns the frequency distribution of tokens in designated categories. Then using this knowledge the classifier can predict classification category for body of texts.

To implement language model classifiers, we used LingPipe[1]. LingPipe is a java based NLP toolkit with wide variety of support for natural language processing tasks and language model based classification and evaluation.

3.8 Question Answering

The experimental question answering module presented in this work answers sentiment based yes/no questions based on the assertions extracted from tweets around a particular context. Each extraction is individually indexed using Lucene[30]. Whenever a question is presented, querying mechanism of Lucene is used to gather all related extractions. Traditional approach to answering yes/no question is based on grouping collected sentences or statements into two groups based on the polarity of the main verb in question [29]. All extractions whose verb matches the polarity go into the yes answer category and the opposite extractions go into the no category.

Our approach is based on sentiment classification. We group collected extractions that match the question presented to the system into different sentiment categories to answer yes/no question.

Chapter 4

4 Experiments and Analysis

4.1 Sentiment Analysis

To assign sentiment values to assertions extracted from tweets, we decided to experiment with well known supervised machine learning techniques, Naive Bayes classifier and Support Vector Machine (SVM). And then we experimented with language model (LM) based classifiers. We experimented with both character LM classifier, as well as token LM or word classifier.

4.1.1 Sentiment Corpus

In order to train our sentiment classifiers, we used a set of manually classified assertions extracted from tweets around the US Presidential election 2012. The collected tweets were first cleaned using the Twitter processing pipeline described in Chapter 3.

We then performed machine translation of the so collected tweets using Moses. After that using the relation extraction methods again explained in Chapter 3, we extracted binary relations from the collected tweets. These extractions were then manually categorized under three sentiment classes, positive, negative and neutral by the help of a group of volunteers by taking into account the nature of sentiment carried by each of these extractions. Our sentiment dataset consists of 2200 positive extractions, 2200 negative extractions and 1800 neutral extractions. All the classified extractions are in the form of relation triplet, (argument1, relation phrase, argument2).

4.1.2 Corpus Analysis

From the so gathered sentiment corpus, known English language stop-words were removed. Stop words do not add to the sentiment score of any statement or sentence and are regarded as noise in sentiment classification. Similarly, known names of political entities, people and organizations were also removed so as to remove any biasness that may be caused by their presence. We then checked the word frequency of our corpus. Figure 4.1 presents a graph with word frequencies plotted. The distribution follows Zipf's law, which is a necessary parameter for a good corpus.

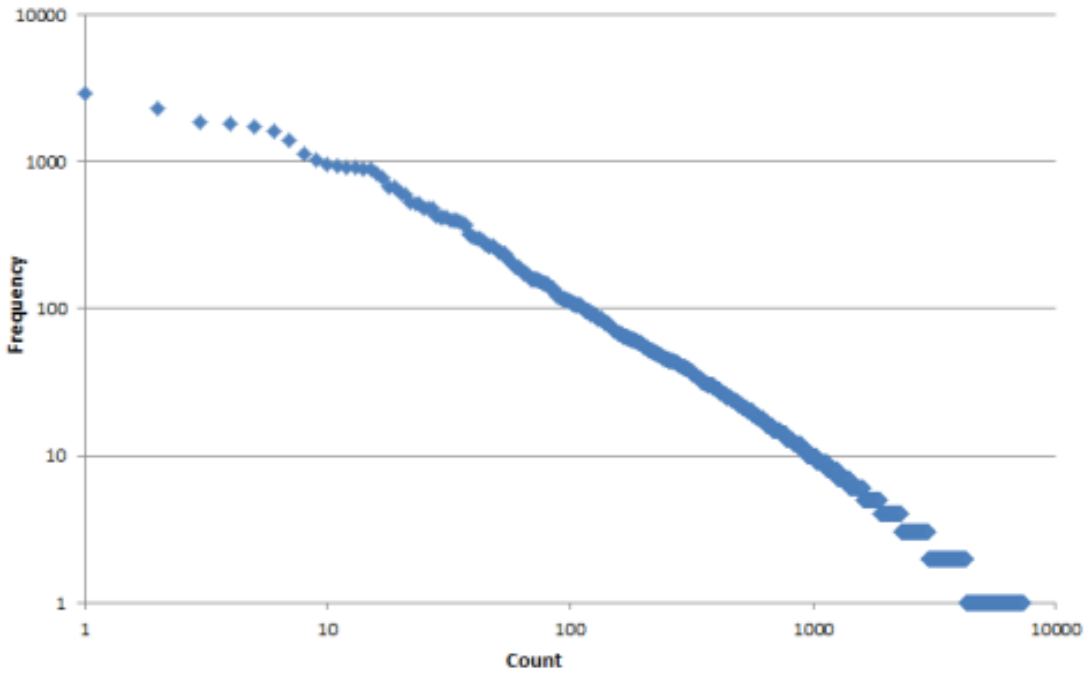

Figure 4.1: Word Frequency

4.2 Sentiment Cloud

From the colleced corpus, we created a sentiment cloud to represent what words make up positive sentiments and what words constitute negative sentiments. The positive wordcloud shown in figure 4.3 represent the most frequent words that appear in our positive corpus. Similarly, figure 4.4 shows us what words or entities are frequent in our negative corpus. These clouds represent our corpus before names of political entities were removed and they in themselves present a general idea of sentiment distribution in the collected corpus.

4.3 Experimental Design

The purpose of the experiments is to compare traditional machine learning based classifiers like Naive Bayes and SVM with modern Language Model(LM) based classifiers. We conducted experiments to compare the performance of n-gram term presence based Naive Bayes classifier, Support Vector Machine(SVM) and character and token based LM classifiers. To classify the collected relations into three categories, positive, negative and neutral using binary classifiers, we used the concept of hierarchical classification with a binary classifier created for each pair of sentiment categories.

4.3.1 Naive Bayes Classifier

To classify relations extracted from Tweets, we first used Naive Bayes method with term unigram as our feature set. Using 5-fold cross validation we achieved an accuracy of 80.00% for two-class positive vs. negative polarity classification. But with neutral vs. positive and neutral vs. negative, this accuracy dropped to 77% and 76.5% respectively. This phenomenon was observed across all classifiers and we realized that most sentences that were labeled neutral still had a high frequency of polar words. It was the construct of the sentences that made the annotator label them neutral. For rest of the experiments we based our classifiers performance on two-class positive vs. negative sentiment classification. With bigrams chosen as feature, the performance of Naive Bayes improved slightly to 80.25%. And when the feature set was the combination of both unigrams and bigrams, the performance improved to 83.75% accuracy. Table 4.1 shows how Naive Bayes classifier performed with our dataset.

Positive vs. Negative			Positive vs. Neutral			Negative vs. Neutral		
Unigram	Bigram	Unigram + Bigram	Unigram	Bigram	Unigram + Bigram	Unigram	Bigram	Unigram + Bigram
80.00%	80.25%	83.75%	77.00%	77.00%	79.25%	76.50%	76.00%	77.77%

Table 4.1: Performance of Naive Bayes Sentiment Classifier

4.3.2 Support Vector Machine (SVM)

SVM performed poorly with our dataset. We achieved an accuracy of 78.89% with two class positive vs. negative sentiment classification with unigrams + bigrams chosen as our feature. This was mainly because our training dataset was small and feature chosen was very sparse for SVMs liking. Table 4.6 shows how SVM performed with our dataset with unigrams + bigrams as feature set.

Positive vs. Negative	Positive vs. Neutral	Neutral vs. Negative
78.89%	70.44%	73.1%

Table 4.2: Performance of SVM

4.3.3 Character N-gram LM Classifier

After SVM, we experimented with character based N-gram LM classifier. For this purpose, we used Lingpipe's LM classifier [1]. It has been shown that Naive Bayes classifier augmented with LMs perform better at sentence classification [38]. Our experiments show that given a small set of training data if the context of sentiment analysis is to be made rather, narrrow, that is limited to certain and fixed area, pure LM classifiers tend to perform better than both Naive Bayes and SVM. With character based method, we achieved an accuracy of 84.54% with an n-gram of size 6. N-grams of size 8 and 10 gave 82.36% and 82.85% accuracy respectively. Table 4.3 shows how the character based LM classifier performed with our dataset.

Positive vs. Negative			Positive vs. Neutral			Negative vs. Neutral		
N-gram = 6	N-gram = 8	N-gram = 10	N-gram = 6	N-gram = 8	N-gram = 10	N-gram = 6	N-gram = 8	N-gram = 10
84.54%	82.36%	82.85%	68.71%	67.79%	68.20%	73.58%	73.84%	73.33%

Table 4.3: Performance of Character N-gram Language Model

4.3.4 Token Language Model Classifier

With token based LM classifier, unigrams as feature gave an accuracy of 80% while with bigrams, the accuracy as somewhat constant at, 80.25% Table 4.4 shows how the token LM classifier performed with our dataset.

Positive vs. Negative		Positive vs. Neutral		Negative vs. Neutral	
Unigram	Bigram	Unigram	Bigram	Unigram	Bigram
80%	80.25%	67%	69.25%	76%	77.77%

Table 4.4: Token N-gram Language Model based Classification

4.4 Comparison

Table 4.5 lists the performance of our selected approach, character LM classifier as compared to some known prior works.

4.5 Question Answering

As we explained in Chapter 3, traditional way of answering yes/no questions from a knowledge base is based on grouping matching answers into two groups based on the

Work	Accuracy
N-gram based topic categorization: Cavnar adn Trenkle (1994)	80%
Pang and Lee (2002) with movie review dataset	82.7% with Unigrams + Bigrams as feature set and SVM as classifier
Pak and Paroubek (2010) with Twitter emoticon based dataset	70%
Go et al.(2009) with emoticon based dataset	82.7% with Unigrams + Bigrams and using Naive Bayes as classifier
Contextualized Character LM Model	**84.54% with n-gram of size 6**

Table 4.5: Performance Comparison with some previous works

polarity of the main verb present in the questin. We have used sentiment classification method to answer sentiment based yes/no question limited to a particular context. For example, for a question like "Will Obama win in Ohio?" or a more generic question line "Is Obama a good president?", our approach follows the following steps to come up with an answer.

- We gather all assertions that have the certain phrases. For the question, "Will Obama win in Ohio?", all assertions that have Obama and Ohio are first collected. In a broader context, this approach will possibly yield a lot of data that have nothing to do with Obama winning or losing in Ohio. But for our purpose, where we have bound our knowledge base very tightly to a particular context, most assertions will relate to Obama and the election in Ohio.

- Next, all the collected assertions are then assigned sentiment values. For the

implementation purpose, we selected the best performing classifier, character LM with n-gram of size 6. To do a three way classification, we just used the positive vs negative classifier with sink approach. We observed that all the candidate extractions that were labeled negative or positive had a very high polar score. We used Lingpipe's [40] scored-classifier to assign score for each classification. Every extraction that received less than 90% confidence in either of the categories, namely positive and negative were grouped into a third sink category, neutral.

- To answer the question, a simple majority idea was then used.

From the contextualized KB created from tweets around the week of April 15-21, 2012, we were able to obtain 123 hits for Obama and Ohio. After sentiment analysis, we obtained the following result.

Yes	No	Neutral
40/123 (32.52%)	47/123 (38.21%)	36/123 (29.26 %)

Table 4.6: Will Obama win in Ohio? (Week of April 15-21, 2012)

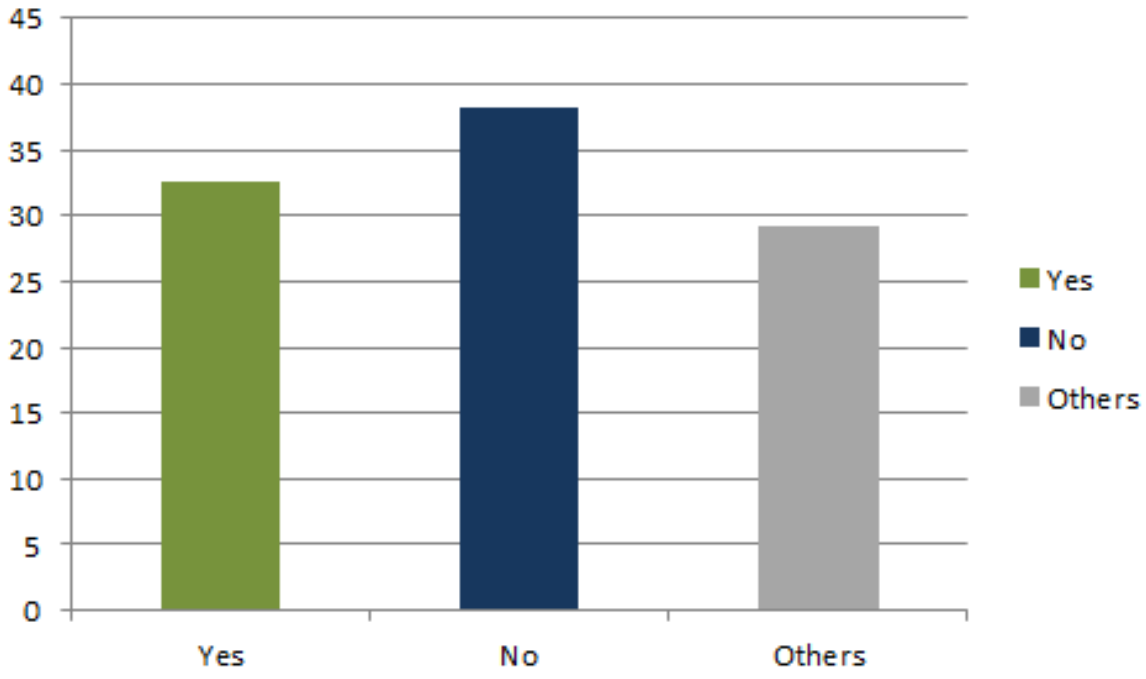

Figure 4.2: Will Obama win in Ohio? (Week of April 15-21, 2012)

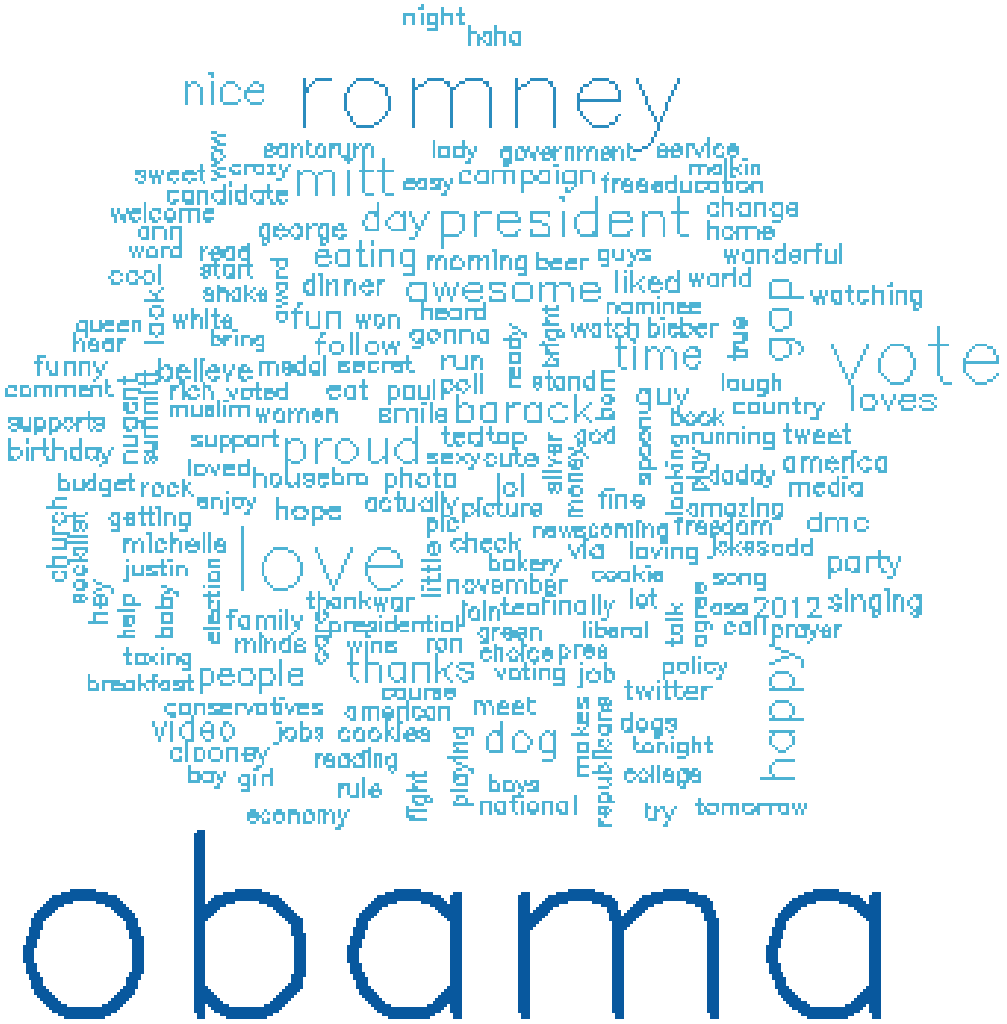

Figure 4.3: Positive Sentiment Word Cloud

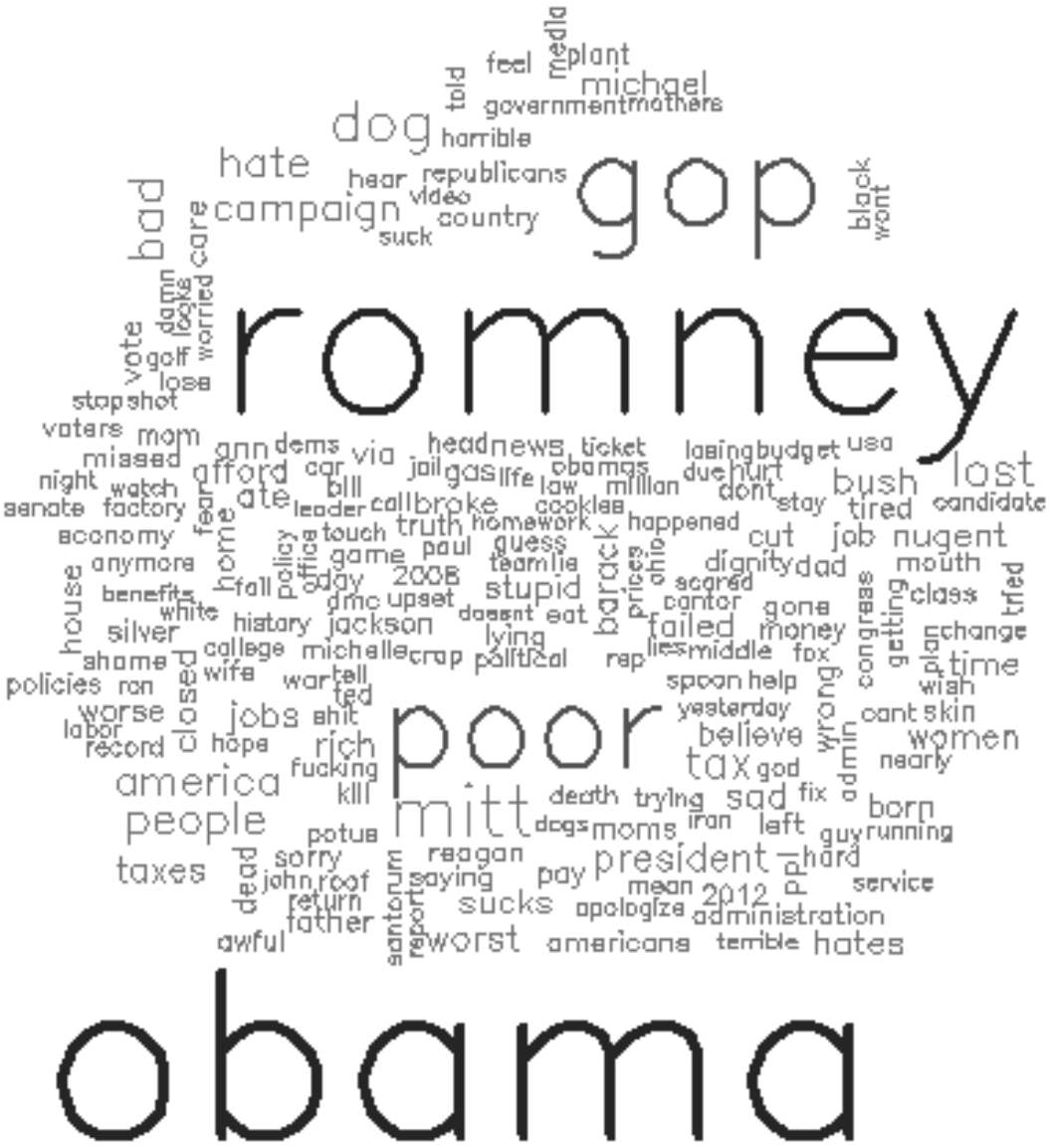

Figure 4.4: Negative Sentiment Word Cloud

Chapter 5

Conclusion and Future Work

In this work we presented a linguistic framework that allows for information extraction from tweets in the form of binary relation triplets, (argument1 relation phrase argument2), with each triplet representing an assertion embedded in a tweet. Taking tweets around the US presidential election 2012 as an example, we extracted assertions around various candidates and political parties. We then demonstrated that Language Model (LM) based classifiers can outperform traditional supervised classifiers like SVM and Naive Bayes in context dependent sentence level text classification. We used a hierarchical classifying method to categorize assertions extracted from tweets into one of these three categories: positive, negative or neutral.

Most sentiment classification work carried out so far are based on a large set of automatically generated training set that is dependent on emoticons present in tweets. For example, all tweets that have positive emoticons are grouped into positive cat-

egory and the tweets that contain negative emoticons and grouped into negative category. Our approach to sentiment analysis is based on a small set of manually labeled training data. Our experiments have shown that for a context dependent text classification, a small set of manually labeled data can perform better than emoticons based training data.

Our approach to sentiment analysis of tweets, in which rather that doing a holistic sentiment analysis of individual tweets, individual assertions are extracted and sentiment analysis is performed on these extractions helps in performing a deep level sentiment analysis of tweets. This model is especially helpful when there are multiple opinions about multiple entities in a single tweet. It makes sense that in such cases rather than assigning a single sentiment label to an entire tweet, we extract individual assertion and assign sentiment value to each one of them separately. And the process of extracting assertions and backing them up with sentiment values also allows us to create a knowledge base that is able to answer contextualized yes/no questions.

5.1 Limitations

The framework we have presented in this work has a number of limitations.

- Our approach to sentiment analysis is heavily coupled to a given context. The training set we have used for sentiment classification in our work, which was created from tweets discussing politics, especially elections, does not perform

well when used in other scenarios.

- Our approach to information extraction from Twitter suffers from a lot of extractions which are incomplete and incoherent.

5.2 Future Work

The scope of this work can be furthered in the following way :

- Machine learning based relation extraction can be applied on Twitter. If information extraction domain is limited, a supervised approach to relation extraction looks feasible.

- Further study of language model based classifiers to classify proper tweets needs to be explored.

- Embedding the knowledge base created from assertions extracted from tweet into known ontological format to create a ontology based question answering system that can give answer to any question related to a given context and news.

- Support for real time analysis of tweets.

- Deploying this approach to tweets about product reviews where product comparison is very common.

Bibliography

[1] Lingpipe 4.1.0. http://alias-i.com/lingpipe. Accessed October 17, 2011.

[2] The r project for statistical computing. http://www.r-project.org. Accessed January 5,2012.

[3] Apache opennlp. http://opennlp.apache.org/, 2006.

[4] Twitter study reveals interesting results about usage. http://www.pearanalytics.com/blog/wp-content/uploads/2010/05/Twitter-Study-August-2009.pdf, August 2009. Accessed February 2,2012.

[5] 200 million tweets per day. http://blog.twitter.com/2011/06/200-million-tweets-per-day.html, June 2011. Accessed January 11,2012.

[6] Streaming api. http://dev.twitter.com/docs/streaming-api, April 2012. Accessed April 30,2012.

[7] Michele Banko, Michael J. Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. Open information extraction from the web. In *In the Proceedings*

of the 20th International Joint Conference on Artificial Intelligence, pages 2670–2676, January 2007.

[8] Michele Banko and Oren Etzioni. The tradeoffs between open and traditional relation extraction. In *Proceedings of ACL-08: HLT*, pages 28–36, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[9] Luciano Barbosa and Junlan Feng. Robust sentiment detection on twitter from biased and noisy data. In *COLING (Posters)*, pages 36–44, 2010.

[10] Mikhail Bautin, Lohit Vijayarenu, and Steven Skiena. International sentiment analysis for news and blogs. In *Proceedings of the International Conference on Weblogs and Social Media (ICWSM)*, 2008.

[11] Albert Bifet, Geoffrey Holmes, Bernhard Pfahringer, and Ricard Gavaldà. Detecting sentiment change in twitter streaming data. *Journal of Machine Learning Research - Proceedings Track*, 17:5–11, 2011.

[12] Christopher J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[13] Deepayan Chakrabarti and Kunal Punera. Event summarization using tweets. In *ICWSM*, 2011.

[14] Michael J. Crawley. *The R book*. Wiley, 2007.

[15] Oren Etzioni, Anthony Fader, Janara Christensen, Stephen Soderland, and Mausam. Open information extraction: The second generation. In *IJCAI*, pages 3–10, 2011.

[16] Anthony Fader, Stephen Soderland, and Oren Etzioni. Identifying relations for open information extraction. In *EMNLP*, pages 1535–1545, 2011.

[17] Tristan Fletcher. Support vector machines explained. 2009.

[18] Michael Gamon. Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, 2004.

[19] Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. Part-of-speech tagging for twitter: Annotation, features, and experiments. In *ACL (Short Papers)*, pages 42–47, 2011.

[20] Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. *Processing*, pages 1–6, 2009.

[21] Open Group. Regular expressions. `http://pubs.opengroup.org/onlinepubs/009695399/basedefs/xbd_chap09.html`, 2004.

[22] Tek Group and Western Kentucky University. Social media news sur-

vey 2011. http://www.tekgroup.com/pdf/2011_Social_Media_News_Survey.pdf, 2011. Accessed March 15,2012.

[23] Ryuichiro Higashinaka, Marilyn Walker, and Rashmi Prasad. Learning to generate naturalistic utterances using reviews in spoken dialogue systems. *ACM Transactions on Speech and Language Processing (TSLP)*, 2007.

[24] Nancy Ide and Catherine Macleod. The american national corpus: A standardized resource for american english. In *Proceedings of Corpus Linguistics 2001*, pages 831–836, 2001.

[25] T. Joachims. SVM light, http://svmlight.joachims.org, 2002.

[26] Holmer Hemsen Kathrin Eichler and Gnter Neumann. Unsupervised relation extraction from web documents. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, may 2008.

[27] M. Kaufmann. Syntactic Normalization of Twitter Messages. *studies*, 2, 2010.

[28] Efthymios Kouloumpis, Theresa Wilson, and Johanna Moore. Twitter sentiment analysis: The good the bad and the omg! In *ICWSM*, 2011.

[29] B. Fortuna D. Mladenic M. Grobelnik L. Dali, D. Rusu. Question answering based on semantic graphs. In *LTC2009*, 2009.

[30] Lucene. The Lucene search engine, 2005.

[31] Andrew Kachites McCallum. Mallet: A machine learning for language toolkit. `http://www.cs.umass.edu/~mccallum/mallet`, 2002.

[32] Yun Niu, Xiaodan Zhu, Jianhua Li, and Graeme Hirst. Analysis of polarity information in medical text. In *Proceedings of the American Medical Informatics Association 2005 Annual Symposium*, 2005.

[33] Alexander Pak and Patrick Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, May 2010.

[34] Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *In Proceedings of the ACL*, pages 271–278, 2004.

[35] Bo Pang and Lillian Lee. Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 115–124, 2005.

[36] Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2007.

[37] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02*

conference on Empirical methods in natural language processing - Volume 10, EMNLP '02, pages 79–86, 2002.

[38] Fuchun Peng, Dale Schuurmans, and Shaojun Wang. Augmenting naive bayes classifiers with statistical language models, 2003.

[39] Rudy Prabowo and Mike Thelwall. Sentiment analysis: A combined approach. *J. Informetrics*, 3(2):143–157, 2009.

[40] K. Raghunathan and S. Krawczyk. CS224N: Investigating SMS Text Normalization using Statistical Machine Translation. 2009.

[41] Juan Ramos. Using tf-idf to determine word relevance in document queries, 1999.

[42] Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. Named entity recognition in tweets: An experimental study. In *EMNLP*, pages 1524–1534, 2011.

[43] Delia Rusu, Lorand Dali, Blaž Fortuna, Marko Grobelnik, and Dunja Mladenić. Triplet extraction from sentences. 2007.

[44] Beaux Sharifi, Mark-Anthony Hutton, and Jugal Kalita. Summarizing microblogs automatically. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 685–688, 2010.

[45] M. Stevenson. An Unsupervised WordNet-based Algorithm for Relation Extraction. In *Proceedings of the Fourth International Conference on Language*

Resources and Evaluation workshop "Beyond Named Entity: Semantic Labelling for NLP tasks", Lisbon, Portugal, 2004.

[46] Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, pages 173–180, 2003.

[47] Twitter.com. Enabling rush of innovation. `http://blog.twitter.com/2010/03/enabling-rush-of-innovation.html`, 2010. Accessed March 17, 2012.

[48] Shaomei Wu, Jake M. Hofman, Winter A. Mason, and Duncan J. Watts. Who says what to whom on twitter. In *Proceedings of the 20th international conference on World wide web*, pages 705–714, 2011.

[49] Kiduk Yang, Ning Yu, Alejandro Valerio, and Hui Zhang. WIDIT in TREC-2006 Blog track. In *Proceedings of TREC*, 2006.