5-24-2013

The Linguistics of Sentiment Analysis

Laurel Hart
Portland State University

The Linguistics of Sentiment Analysis

by

Laurel Hart

An undergraduate honors thesis submitted in partial fulfillment of the

requirements for the degree of

Bachelor of Arts

in

University Honors

and

Applied Linguistics

Thesis Adviser

George Tucker Childs

Portland State University

2013

Abstract

Computational linguistics is a field that was founded by linguists, but more recently is the domain of more computer scientists than linguists. Use of data-driven and machine learning methods for computational linguistics applications is now more common than hand-written linguistic rules. In order for a linguist to enter the field, it is essential that he or she be familiar with methods and techniques from computer science. The purpose of this paper is two-fold. The first is to serve as a linguist's introduction to concepts from outside of linguistics that are used in computational linguistics. The second purpose is to illustrate the use of linguistic features for a specific task known as sentiment analysis. This task involves determining the sentiment of a piece of text. By way of examining linguistics within sentiment analysis, this paper will begin to gesture at the potential role for linguists in the modern field of computational linguistics as a whole. The goal is to encourage and enable linguists to re-engage with computational linguistics by providing a suitable introductory work.

Table of Contents

Part I
Introduction

Though this paper is titled *The Linguistics of Sentiment Analysis,* an equally accurate alternate title would be *Sentiment Analysis for Linguists.* It is undeniable that most of the work in the field of sentiment analysis—and even more broadly, in the field of computational linguistics—is not only carried out but also written for an audience more specialized in computer science and statistics than in linguistics. Even one of the most extensive and widely-referenced surveys of the field of sentiment analysis leaves discussion of linguistic contributions to the field at "...we were directly charged to focus on information-access applications, as opposed to work of more purely linguistic interest. We stress that the importance of work in the latter vein is absolutely not in question,"[1] and says little more on the subject.

The purpose of this paper is not to show that focusing on the computational side of the field is in any way wrong, but to begin to re-access a field which has become separated from other studies of language. Over the course of its development, the field of computational linguistics has shifted its emphasis from linguistics to computer science—today strides in computational linguistics are made not by linguistic discoveries, but by computational optimizations and exploration of statistical properties. Currently, it is nearly impossible for a linguist with no computer science background to participate in computational linguistics, but quite commonplace for a computer scientist with little knowledge of linguistics to participate. This field, which was formerly dominated by linguists, has become so steeped in specialized, technical language that is not approachable without equally-specialized training.

The contribution of linguistics is "not in question," but the role of linguists in modern computational linguistics is less certain than ever. By way of examining linguistics within sentiment analysis, this paper will begin to gesture at the potential role for linguists in the field of computational linguistics as a whole, and hopes to encourage linguists to re-engage with computational linguistics by providing a suitable introductory work.

But first, some caveats:

For the sake of space, this paper will deal only with Natural Language Processing and Sentiment

1 (Pang & Lee, 2008)

Analysis in English. Other languages present different—but interesting!—challenges that would be glossed over at best if included here.

Additionally, this paper is concerned with textual data, rather than audio/verbal data, and in particular mostly that available on the World Wide Web (web). This focus follows current trends in sentiment analysis research.

This paper is organized as follows: Part II lays a groundwork for discussing sentiment analysis by giving definitions for its base components. These include algorithms, machine learning, natural language processing, linguistic features, and methods for measuring the success of a sentiment analysis system. Though the focus is on the wider topics of Computational Linguistics and Natural Language Processing, this section covers only topics relevant to sentiment analysis. Part III delves more specifically into sentiment analysis, utilizing the components introduced in Part II. Part III includes a general description of sentiment analysis, in addition to three specific tasks within sentiment analysis. Part IV concludes with some final thoughts on sentiment analysis and computational linguistics.

Part II
Groundwork

1 *Computational Linguistics Natural Language Processing*

Computational Linguistics (**Comp Ling**) arose in the 1950s with the first efforts at automated translation[2], though it was not known by this name until the mid-1960s.[3] Linguists of the time thought that the potential of mechanizing linguistics lay "less in the possibility of deriving a characterization of the translation relation from emergent properties of parallel corpora, than in carrying out exactly, and with great speed, the minutely specified rules that they would write." [4] In other words, linguists expected to be the ones to discover and enumerate the facts of language, and computers only to assist in abiding by them. Since then, the field of Comp Ling has expanded, and now encompasses much more than just the original task of machine translation; other areas of research include corpus linguistics, computational semantics, creation of analytical systems for parsing and part-of-speech tagging, speech recognition, speech synthesis, and text summarization. Because many of these also fall under Natural Language Processing (**NLP**), the two terms are sometimes used more or less synonymously. Contemporary NLP is closely related to the fields of Artificial Intelligent (**AI**), Information Retrieval (**IR**), Information Extraction (**IE**). Techniques for these fields often overlap and draw upon one another.

Approaches to NLP can very loosely be divided into two groups: hand-coded rules and data-driven methods, which will be covered in more detail below.[5] As with Comp Ling, NLP has its origins in the former, but now utilizes the latter almost exclusively. The transition from one to the other was largely influenced by advancements in the field of AI, the proliferation of online linguistic data, and heightened public availability of that data.[6]

Because linguistic data is so crucial to modern NLP systems, the following sections will examine

2 More commonly known as machine translation or mechanical translation.
3 (Mitkov, 2005)
4 (Mitkov, 2005)
5 Exceptions to this grouping probably exist, but this paper focuses on the machine learning set.
6 (Pang & Lee, 2008)

the compilation of a corpus in detail, including characterizations of three increasingly common but non-traditional sources of linguistic data. This examination will be followed by explanation of how the data is incorporated into an NLP system—a less straight-forward process than one might expect.

1.1 Building a corpus for NLP

Quite naturally, the main type of data that Comp Ling is concerned with is extracts of language, typically textual. Traditionally, linguistic corpora for English have been compiled from sources such as the *Brown University Standard Corpus of Present-Day American English*, the *Wall Street Journal Corpus*, and the Penn Treebank, a parsed corpus which includes both.[7] These sources are comprised of newspapers, and are generally used to represent a "standard" variety of written English. These sources can be said to belong solidly to only one relatively formal register of English, and so cannot be taken for accurate representations of less formal registers such as conversational speech or writing, more formal registers such as academic writing, or simply different registers such as fictional novels. Such corpora are similarly inappropriate for NLP systems which are concerned with text from the web, where non-standard linguistic forms abound.

As (Crystal, 2011) puts it, the web is the "largest database of language the world has ever seen" so naturally, "we would expect to find linguists exploring it, to see what is going on." Though NLP is in no way constrained to text on the internet, linguists can hardly overlook such a cornucopia of linguistic data.

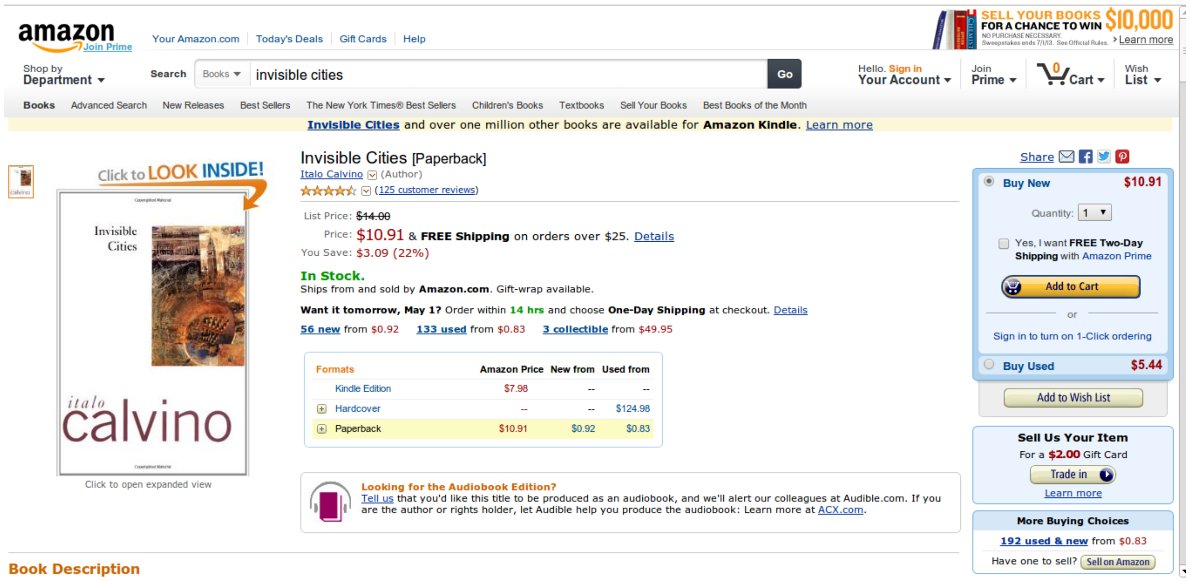
NLP tends to use web data not only because the data is abundant, but because there is interest in developing applications for textual forms seen *only* on the web. This includes tasks like filtering spam emails and tracking opinions in real time as they are published on the internet. Further examples will be given in Part III.

The creation of a new corpus, or new corpora, is required to begin modeling the language of the web; luckily, the exact advantages of gathering online data are its abundance and ease of access. The next three sections detail the linguistically unique properties of three increasingly common sources of linguistic data for NLP. The following sections will demonstrate the importance of familiarity with the unique linguistic features of any particular medium.

1.1.1 Amazon

Amazon is an online marketplace, where one can search for just about any kind of product. A typical product page resembles Figure 1. It features the product name, manufacturer, a picture, the price, and so on.

7 (Mitkov, 2005) (Marcus et al., 1999)



Invisible Cities and over one million other books are available for **Amazon Kindle**. Learn more

Invisible Cities [Paperback]
Italo Calvino (Author)
(125 customer reviews)

List Price: ~~$14.00~~
Price: $10.91 & **FREE Shipping** on orders over $25. Details
You Save: $3.09 (22%)

In Stock.
Ships from and sold by **Amazon.com**. Gift-wrap available.

Want it tomorrow, May 1? Order within **14 hrs** and choose **One-Day Shipping** at checkout. Details

56 new from $0.92 133 used from $0.83 3 collectible from $49.95

Formats	Amazon Price	New from	Used from
Kindle Edition	$7.98	--	--
Hardcover	--	--	$124.98
Paperback	$10.91	$0.92	$0.83

Looking for the Audiobook Edition?
Tell us that you'd like this title to be produced as an audiobook, and we'll alert our colleagues at Audible.com. If you are the author or rights holder, let Audible help you produce the audiobook: Learn more at ACX.com.

Buy New $10.91
Quantity: 1
Yes, I want **FREE Two-Day Shipping** with Amazon Prime
Add to Cart
or
Sign in to turn on 1-Click ordering
Buy Used $5.44
Add to Wish List

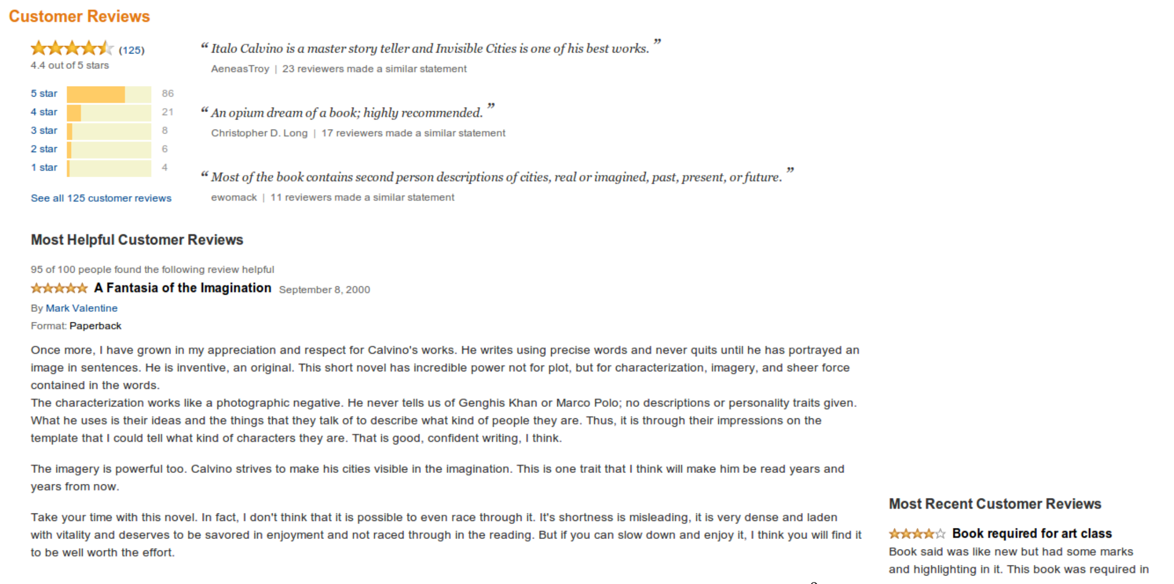
Sell Us Your Item
For a **$2.00** Gift Card
Trade in
Learn more

More Buying Choices
192 used & new from $0.83
Have one to sell? Sell on Amazon

Book Description

Figure 1: A typical product page on amazon.com.[8]

Typically of more interest to Comp Ling, however, is the bottom of the page, as seen in Figure 2: the customer review section.



Customer Reviews

(125)
4.4 out of 5 stars

5 star	86
4 star	21
3 star	8
2 star	6
1 star	4

See all 125 customer reviews

“*Italo Calvino is a master story teller and Invisible Cities is one of his best works.*”
AeneasTroy | 23 reviewers made a similar statement

“*An opium dream of a book; highly recommended.*”
Christopher D. Long | 17 reviewers made a similar statement

“*Most of the book contains second person descriptions of cities, real or imagined, past, present, or future.*”
ewomack | 11 reviewers made a similar statement

Most Helpful Customer Reviews

95 of 100 people found the following review helpful
A Fantasia of the Imagination September 8, 2000
By Mark Valentine
Format: Paperback

Once more, I have grown in my appreciation and respect for Calvino's works. He writes using precise words and never quits until he has portrayed an image in sentences. He is inventive, an original. This short novel has incredible power not for plot, but for characterization, imagery, and sheer force contained in the words.
The characterization works like a photographic negative. He never tells us of Genghis Khan or Marco Polo; no descriptions or personality traits given. What he uses is their ideas and the things that they talk of to describe what kind of people they are. Thus, it is through their impressions on the template that I could tell what kind of characters they are. That is good, confident writing, I think.

The imagery is powerful too. Calvino strives to make his cities visible in the imagination. This is one trait that I think will make him be read years and years from now.

Take your time with this novel. In fact, I don't think that it is possible to even race through it. It's shortness is misleading, it is very dense and laden with vitality and deserves to be savored in enjoyment and not raced through in the reading. But if you can slow down and enjoy it, I think you will find it to be well worth the effort.

Most Recent Customer Reviews

Book required for art class
Book said was like new but had some marks and highlighting in it. This book was required in

Figure 2: At the bottom of the product page: customer reviews.[9]

This is where people who have purchased the relevant product—through Amazon or otherwise—give opinions about it, typically to help guide other users to make a purchasing decision. The text of good

8 (amazon.com, 2013)
9 (amazon.com, 2013)

Amazon reviews has been described as more structured than that found on other parts of the web.[10] Although there is no strict requirement, text typically follows a formal structure. Features that make it distinct from writing in other contexts, such as writing in all capitalized letters, are much less frequent than in other web media.

The reviews give at least a few basic pieces of information: a rating out of five stars, the name of the reviewer, a title for the review, and the text of the review. The target of the review is reliably the product displayed on the page, though sometimes other, related products and services may be mentioned: comparable products, the quality of service from the seller and shipper. Amazon allows users to sort reviews by the number of stars given; this can be used to find common complaints about or praises of the product.

Data from Amazon is used in (Danescu-Niculescu-Mizil, Kossinets, Kleinberg, & Lee, 2009), (Tsur, Rappoport, & Davidov, 2010), and (Davidov et al., 2010), which are discussed in Part III, Section 2.3.

1.1.2 Livejournal

Livejournal allows its users to create web logs, or "blogs." These can address any topic, from baking to computer programming to fan fiction to social justice. Posts are given a date, title, and have an optional field for entering the author's current mood. It is the current mood option that makes this source particularly attractive for certain types of sentiment analysis. The user is given a choice of 132 common moods, including items such as “angry” and “amused,” and has the option of entering a new mood.[11]

Table 1: Example emoticons

Emoticon	Emotion represented
:)	smiling, happiness (vertical) less commonly (:
^_^	happiness (horizontal)
:(	sadness, unhappiness (vertical) less commonly):
(╯°□°）╯︵ ┻━┻)	“flipping tables” extreme anger
ಠ_ಠ	“look of disapproval”

Additional expression of mood can be found in the form of “emoticons,” or textual representations of facial expressions, as appear Table 1. These are written into the blog entries themselves, and may be used to clarify the sentiment of an otherwise emotionally-ambiguous sentence, such as “About to get my hair redone :)” Some emoticons are simple, constructed from punctuation found on a typical QWERTY keyboard. Others are quite complicated and make use of multi-language fonts. These are found not just on Livejournal, but on media across the internet.

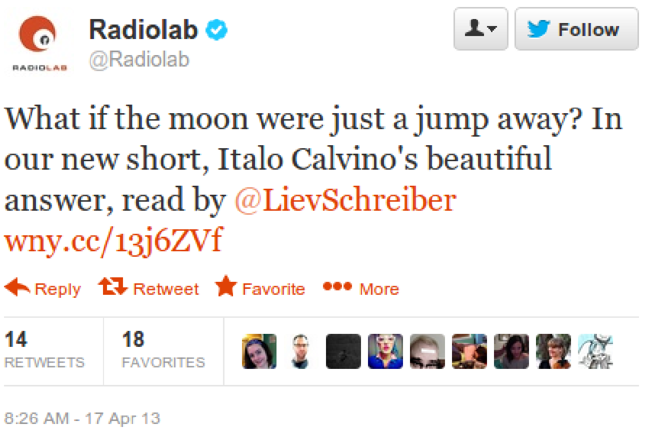
The text of a Livejournal entry can vary widely from author to author in both style and length. While some authors' entries may be lengthy, well-structured, and written in a standard English variety, others choose to publish short notes, possibly leave off punctuation, use emoticons, sometimes write in an “accent” (*e.g.*, “wif” for “with”), use abbreviations common to the web, such as “btw” for “by the way,” “lol” for “laugh out loud,” “irl” for “in real life,” (as opposed to online life) and so on. Any combination of these features may exist. The Livejournal platform also allows for comments and replies, but these are less typically used for corpus building.

(Mishne, 2005) and (Généreux & Evans, 2006) in Part III, Section 2.2 use data from Livejournal.

10 (Davidov, Tsur, & Rappoport, 2010)
11 (Mishne, 2005)

1.1.3 Twitter

Twitter is a "microblogging" platform which limits individual entries to 140 characters.[12] These short selections of text, called "tweets" are shared with an audience, called "followers." The act of posting a tweet to the web is called "tweeting."


Radiolab
@Radiolab
Follow

What if the moon were just a jump away? In our new short, Italo Calvino's beautiful answer, read by @LievSchreiber wny.cc/13j6ZVf

Reply Retweet Favorite More

14 RETWEETS 18 FAVORITES

8:26 AM - 17 Apr 13

Figure 3: Example tweet courtesy of @Radiolab.[13] [14]

There are myriad features which make tweets unique from other types of text. Length is an obvious one, but it also brings about another phenomenon: causing, for example, shortening words as much as possible so as to save on characters. Similar abbreviation can be observed throughout the web for the simple reason of shorthand, but it is much more prevalent on Twitter because of the character limitation.

Twitter has also encouraged several features which are either totally unique or popularized by the platform.

One such practice is to repeat of someone else's tweet and pass it on to one's own followers or bring it to another tweeter's attention. This is called "retweeting," and obsure who the original tweeter was. To combat this, retweeters add "RT" and the original username to the message, as in Figure 4.

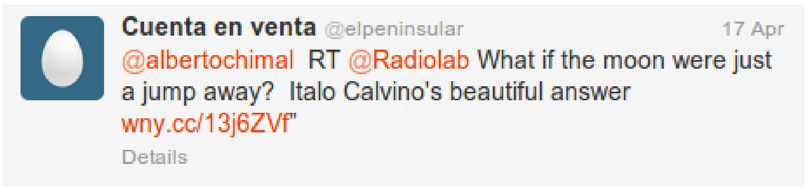

Cuenta en venta @elpeninsular 17 Apr
@albertochimal RT @Radiolab What if the moon were just a jump away? Italo Calvino's beautiful answer wny.cc/13j6ZVf"
Details

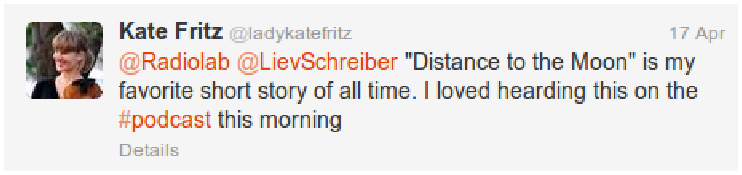
Figure 4: RT in action.[15]

Tweeters refer to one another by using the form @username. The Twitter website automatically links to the user's own Twitter page when linked using that format, sometimes called the "target."[16] Tweeters can also use this format to address tweets to one another. In Figure 4, @elpeninsular retweeted @Radiolab's tweet,[17] addressing it to @albertochimal.

12 Meaning letters, numbers, punctuation, and spaces.
13 (Radiolab, 2013)
14 Although it may not look like it, the last part of this tweet is a link. Twitter automatically shortens links so that they do not use up too many characters.
15 (Elpeninsular, 2013)
16 (Agarwal, Xie, Vovsha, Rambow, & Passonneau, 2011)
17 With some changes, but likely to take back enough characters to address it and add "RT."

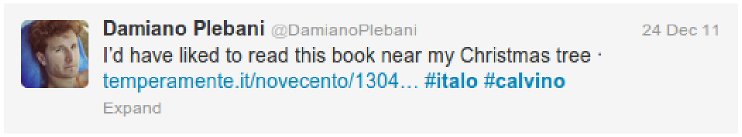
Besides following individual tweeters, it is also possible to browse tweets by topic. This is done by searching for a "hashtag" or "tag," shown in the form of #topic. On Twitter, multi-word hashtags are written without spaces, usually with camel capitalization, e.g., #ThisIsAHashtag. Hashtags can help to identify the topic or characteristic of a tweet if it is not obvious, making it possible to address a topic without giving full context in the tweet itself. They can appear as part of the message, as in Figure 5a, though they are more typically placed at the end of a message as in Figure 5b.

a



Kate Fritz @ladykatefritz 17 Apr
@Radiolab @LievSchreiber "Distance to the Moon" is my favorite short story of all time. I loved hearding this on the #podcast this morning
Details

b



Damiano Plebani @DamianoPlebani 24 Dec 11
I'd have liked to read this book near my Christmas tree · temperamente.it/novecento/1304… #italo #calvino
Expand

Figure 5: #example #tweet[18] [19]

Hashtags are a famous and infamous part of the Twitter platform that allows its users to easily track "trending" topics, be they the latest music or events that have happened within minutes, often before they are picked up by news channels.

But why is it worth knowing these things about the Twitter platform?

It is important to consider these features because, without management, they can skew a dataset. For example, a particularly popular tweeter may be retweeted by millions of followers and followers of followers. This means that one piece of text is repeated millions of times with little variation and little input from the "speakers" repeating it. One might handle this by simply removing all tweets that use the shortening "RT" in them. This may cause some tweets—ones that use "RT" as abbreviation for something other than "retweet"—to be "wrongfully" removed from the dataset tweets, but likely removes more unwanted, repetitious tweets than desired ones.

Additionally, because of the authorship/username system, the @ and # symbols appears many more times in a Twitter corpus than it would for any other. In order to deal with these Twitter-specific features, decisions must be made how to do so that will certainly affect the quality of the dataset. Simply stripping the # symbol off of hashtags will give some tweets seemingly random words at the end, as in Figure 5b. On the other hand, removing the hashtags altogether will remove important information from some tweets, as in Figure 5a.

These are just some basic factors to consider about the Twitter corpus; other challenges undoubtedly arise.

Data from Twitter is used in (Go & Huang, 2009), (Davidov et al., 2010), (Agarwal et al., 2011), (González-Ibáñez et al., 2011), and (Wang, Can, Kazemzadeh, Bar, & Narayanan, 2012) detailed in Part III.

18 (Fritz, 2013)
19 (Plebani, 2011)

1.2 Pre-processing steps

In any NLP application, one of the first steps is to pre-process the textual data. There are a few common processes, one of the most basic being tokenization or word segmentation. These can be thought of as breaking the text into smaller pieces, often into units such as words. Since spaces are placed between words in English, breaking text into word tokens might seem unchallenging; in most cases it is. Exceptions include multi-word units, such as "San Francisco," and words that contain non-alphabetical characters, such as "non-alphabetical." These kinds of words require special handling to tokenize correctly.

Beyond word-level tokenization, there is also the process of segmenting a document by its sentences, known as sentence splitting. This can typically be done by looking for punctuation such as a period, exclamation mark, or question mark. Important to remember, however, is that not every period is sentence-finial. Periods are often used to note abbreviations, such as "Dr." or "U.S.A.." It may also appear in a numerical setting, as in "$1.50" or "4.5 billion people." Again, the simple solution is likely to result in mistakes, so care must be given to handle these exceptions.

Another particularly important part of pre-processing the data is normalizing the text—that is, putting the text into a form that is consistent both internally and between documents. Consider the appearance of numbers in a text. Some authors may use numerals, others use the written-out form, and still others may use a combination of the two.[20] It probably will not make a difference to the application whether the numbers are all numerals or all written out, as long as the form is consistent such that "ten" and "10" are treated identically. Similarly, words capitalized at the beginning of a sentence may be uncapitalized to match their intra-sentence equivalents, hyphenated words should be made to match their unhyphenated equivalents, and so on.[21] Adding consistency will allow different pieces of text to be processed comparably by the application. Normalizing text is particularly important in IR, where user queries are matched to documents, but is common among other types of Comp Ling applications.

Table 2: Common text normalizations

Numerals	Capitalization	Hyphenation	Diacritics	Initialisms
10	UPPERCASE	un-hyphenated	àçčéñŧëď	T.V.
ten	Uppercase	unhyphenated	accented	TV
	uppercase			

Sometimes the task of normalizing text is more complex than simply changing all instances of "ten" to "10." Reconsider the capitalization problem. In the sentence

Boring minutes made up the boring hours spent waiting at the doctor's office.

the two appearances of the word "boring" should be made to match. However, in the sentence

The fourth-grader's essay on the town of Boring, Oregon began, "Boring is the perfect word to describe this place."

each occurrence of "Boring" refers to a unique sense of "Boring." Some Comp Ling applications need to meaningfully contrast "Boring" and "boring." This includes web search; one does not want to search for "Boring Oregon" and be given results for "boring Oregon," too. So we see that the task of bringing consistency to textual data can be quite complicated.

The final types of textual pre-processing that will be mentioned here are lemmatization and

20 The author was taught to spell out numbers below 10.
21 (Manning, Raghavan, & Schütze, 2009)

stemming. The two are quite similar, but not quite identical. Lemmatization refers to changing a word to its lemma, or "dictionary form." Sometimes the lemma can be slightly different from the given form, like "ran" becoming "run." The relationship of word form to lemma is not always predictable, particularly in a programmatic sense. Stemming, on the other hand, merely shaves affixes off of words in a systematic way. This is a much simpler, more predictable process, but can end up with stranger results. Stemming is also highly dependent upon implementation. Two popular stemmers are the Porter Stemmer by Martin Porter[22] and the Lancaster Stemmer by Chris Paice and Gareth Husk, examples of which included below.

Table 3: Demonstration of lemmatization and stemming

Original sentence[23]	What makes Argia different from other cities is that it has earth instead of air
Lemma form	what make Argia different from other city be that it have earth instead of air
Porter stemmer[24]	what make Argia differ from other citi is ha earth instead of air
Lancaster stemmer[24]	what mak arg diff from oth city is has ear instead of air

1.3 Building a corpus, continued

Why use these as sources when each brings with it so many difficulties and has to be pre-processed? For one, it is very easy to get a *lot* of data from any one of them—this is good for data-driven techniques. For another, one may want to create applications specific to those platforms, in which case the application itself will need to know how to handle incoming, unseen data; a pre-processing module must be created and tailored to the platform's linguistic features.

2 Features, models, and tools of NLP

After a corpus has been compiled and the text has been processed, the next step is to extract features from the pieces of text. That is to say, this step serves to characterize the texts by features which are selected by the researchers. The following section details commonly-used linguistic features and models, and the tools used to extract them.

2.1 Unigrams and Bag of Words (BoW) models

For a particular piece of text, the individual words present in that text can be considered features called "unigrams." Two-word units are called "bigrams" and *n*-word units are called "*n*-grams." In a BoW model, they are represented in an unordered collection (seen below in alphabetical order for convenience). Depending on the application, it may also be useful to record the frequency (number of occurrences) of each word.

Text 1: Marco Polo describes a bridge, stone by stone.[25]

22 (Porter, 2006)
23 (Calvino, 1978)
24 (Bird et al., 2009) implementation used for Porter and Lancaster stemmers.
25 (Calvino, 1978)

BoW representations completely discard the word order and grammar of a text. Some merely record which words occur, while others may add aspects like a frequency count corresponding to each word. In the text above, all but one word occur just once. "Stone" appears twice. The BoW representation of Text 1 is given in Table 4.

By common practice and for reasons of space, Text 1 and further BoW representations are given in a list format:

{ "a":1, "bridge":1, "by":1, "describes":1, "Marco":1, "Polo":1, "stone":2 }

Performing NLP on just one text is pretty trivial. To really make use of the BoW model, it should be applied to a *collection* of texts. To keep things simple, we will start by building our collection by one more text.

Table 4: BoW of Text 1

Word	Frequency
a	1
bridge	1
by	1
describes	1
Marco	1
Polo	1
stone	2

Text 2: “But which is the stone that supports the bridge?” Kublai Khan asks.[26]

BoW representation: { "asks":1, "bridge":1, "but":1, "is":1, "Khan":1, "Kublai":1, "stone":1, "supports":1, "that":1, "the":2, "which":1 }

Although each individual text can be represented as its own BoW, it is generally more useful to represent each text as a subset of the total of all the words that appear in a collection of documents (a corpus). This can be considered a "master list" of all the words that appear in all the documents, and the total number of times each word appears.

{ "a":1, "asks":1, "bridge":2, "but":1, "by":1, "describes":1, "is":1, "Khan":1, "Kublai":1, "Marco":1, "Polo":1, "stone":3, "supports":1, "that":1, "the":2, "which":1 }

Each individual text can then be represented with relation to this collection of words. Words that do not appear in a particular text are given a 0 frequency count.

Text 1: { "a":1, "asks":0, "bridge":1, "but":0, "by":1, "describes":1, "is":0, "Khan":0, "Kublai":0, "Marco":1, "Polo":1, "stone":2, "supports":0, "that":0, "the":0, "which":0 }

Text 2: { "a":0, "asks":1, "bridge":1, "but":1, "by":0, "describes":0, "is":1, "Khan":1, "Kublai":1, "Marco":0, "Polo":0, "stone":1, "supports":1, "that":1, "the":2, "which":1 }

At this point, the program rarely needs to record which specific word is used, so each can be simplified to the frequencies. Consider Table 5.

Table 5: Word frequencies

	a	asks	bridge	but	by	describes	is	Khan	Kublai
Text 1	1	0	1	0	1	1	0	0	0
Text 2	0	1	1	1	0	0	1	1	1

	Marco	Polo	stone	supports	that	the	which
Text 1	1	1	2	0	0	0	0
Text 2	0	0	1	1	1	2	1

26 (Calvino, 1978)

Identical information represented in list format:

Text 1: [1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1, 2, 0, 0, 0, 0]

Text 2: [0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 2, 1]

Obtaining a BoW representation of a document or an entire corpus is particularly simple because it relies only on the information already present in the text: the words. More sophisticated features require tools outside the original document in order to extract its features.

2.2 Named Entities

As mentioned briefly in Section 1.2, multi-word units such as “San Francisco” require special handling. Named Entity Recognition, also known as “name identification and classification,” and “name tagging,” refers to the task of identifying proper nouns, including single- and multi-word units. Identifying them as proper nouns may assist proper linguistic analysis more than counting them as individual words. The presence of Named Entities may be used as a feature itself.

Results for this and other tasks will be shown annotated using Standard Generalized Markup Language (**SGML**), demonstrated in Table 6.[27]

Table 6: Named Entities in Text 1 and Text 2 annotated with SGML

Text 1	Marco Polo describes a bridge, stone by stone.
Annotated	<NAME TYPE=PERSON>Marco Polo</NAME> describes a bridge, stone by stone.
Text 2	“But which is the stone that supports the bridge?” Kublai Khan asks.
Annotated	“But which is the stone that supports the bridge?” <NAME TYPE= PERSON>Kublai Khan</NAME> asks.

There are multiple ways to build a name tagger. A simple but low-performance method involves hand-coding many rules such as “two capitalized words in a row → NAME” “'Dr.' followed by a capitalized word → NAME” and so on. This may also need to include priorities of rules, so the system can “decide” which rule to follow if the rules present conflicts. Compiling a list of such rules can be quite laborious.

Another approach is to give the name tagger a set of word lists, including common names, companies, and locations. Alone, this system would only recognize names from those lists. However, word lists can be combined with mechanisms to recognize patterns and aliases for the same entity, e.g., “Marco Polo” and “Mr. Polo.”

Finally, a name tagger can also be automatically trained using large amounts of (usually hand-annotated) data and techniques introduced in Section 4.

27 (Mitkov, 2005)

Table 7: Sample part-of-speech abbreviations[28]

Abbreviation	Part of speech
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
DT	Determiner

Abbreviation	Part of speech
VB	Verb, base form
VBD	Verb, past tense
VBZ	Verb, 3rd person singular present
IN	Preposition or subordinating conjunction

2.3 Part-of-speech tagging

Tagging parts of speech (**POS**) is another common feature used in NLP. Like name tagging, POS tagging is both a feature and a task in and of itself. It is often a subcomponent of more complex NLP systems. POS taggers are also another example of a system which can be designed using multiple approaches.

The generalized method is to create a dictionary, or lexicon, of words which include possible parts of speech, a "guesser" for unseen words, and a module for "ambiguity resolution" or "disambiguation," which decides between POS alternatives when more than one is possible for a given word.[29]

The ambiguity resolution module is where the variation in approach typically comes in to the POS task. It can be handled using hand-written disambiguation grammars or by using statistical and machine learning methods covered below.

Table 8: POS tagging in Text 1

Text 1	Marco Polo describes a bridge, stone by stone.
Tagged by NLTK[30]	[('Marco, 'NNP'), ('Polo', 'NNP'), ('describes', 'VBZ'), ('a', 'DT'), ('bridge', 'NN'), (',', ','), ('stone', 'NN'), ('by', 'IN'), ('stone', 'NN'), ('.', '.')]
Tagged by Stanford Parser[31]	Marco/NNP Polo/NNP describes/VBZ a/DT bridge/NN ,/, stone/NN by/IN stone/NN ./.

The annotation of Text 1 given in Table 8 may look more familiar in tree form, as in Figure 6.

28 (Marcus et al., 1999)
29 (Mitkov, 2005)
30 (Bird, Loper, & Klein, 2009)
31 (Klein & Manning, 2003)
32 Tree generated using (Klein & Manning, 2003)

2.4 Pattern extraction

Once POS tagging has been performed on a piece of text, patterns in terms of POSs can also be extracted. Sometimes individual word POS tags are discarded in favor of a pattern that represents an entire sentence or phrase. For certain applications, these may be in relation to the predicate, or verb.

Potential patters extracted from Text 1 are given in Table 9.

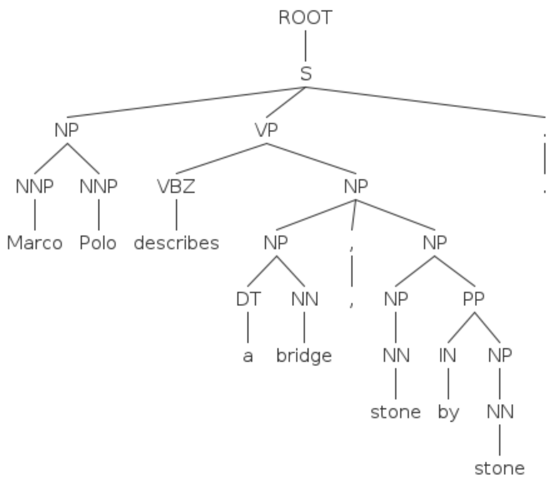


Figure 6: POS annotation in tree form.[32]

Table 9: Patterns extracted from Text 1

Pattern level	Pattern
Sentence	NNP NNP VBZ DT NN NN IN NN
Phrase	NNP NNP VBZ
Phrase	VBZ DT NN NN IN NN
Phrase	NNP NNP VBZ DT NN
Phrase	NN IN NN

3 *Algorithms and Measuring Success*

Simply put, an algorithm is a set of steps. Many—but not all—algorithms produce some kind of output, or result. Different algorithms may generate identical results, or different results toward the same goal. In the former case, the algorithms are typically judged for some desired trait, such as operational efficiency; in the latter, they are compared by success at performing the task; in most cases, algorithms are judged by a mixture of both criteria.

By way of example, take the goal of making a cake. Most cake recipes will follow the general form:

1. Measure ingredients
2. Combine ingredients
3. Put in container
4. Bake
5. Remove from oven after a certain amount of time.

One recipe may specify to mix dry ingredients and wet ingredients separately before combining; another may include the step of sifting the flour. Even though the ingredients are the same, the cakes will come out differently. Why else include the extra steps?

Alternatively, consider altering the *input*—that is, the ingredients—of the recipe. They may vary very little; just the ratio of flour to oil, or slightly more chocolate in one than the other. The end results are different cakes, and the recipes can be judged and ranked by certain features of the produced cake. These features might include flavor, texture, and shape, as appropriate to the particular type of cake desired.

Similarly, in order to give measurements of success for otherwise quite disparate applications, there are some measures that are commonly used in NLP and IE. In order to discuss them, we will talk about them in terms of testing against a corpus made up of documents. This corpus will contain positives—that is, the documents we are attempting to retrieve—and negatives—documents in which we are not interested. Notably, these particular measures can only be taken when testing on a known set; that is, certain information is known about the dataset including the total number of documents, the number of positives, and which ones they are. This information is often hand-annotated on a test-set.

Precision: The number of positive results out of the total number of results.

Recall: The number of positives results out of the total number of positives.

These two terms may be more familiar to the reader in terms of "false positives" and "false negatives"; precision means the minimization of false positive results and recall is the minimization of false negatives results.

Neither of these alone is a very good indicator of success, in exactly the same way that judging a cake by appearance or texture alone fails to fully characterize the cake. While either precision or recall may be of interest in a particular application, **F-score**[33] is a statistical measure that combines the two into one, giving an overall score of accuracy.

All of the above measurements depend on knowing the intended results, which requires the annotation of a “gold-standard” answer key.[34] The results of an NLP system are typically also compared to a baseline, or “default” expectation of performance. The baseline is sometimes defined by the statistical chance of a correct answer when guessing arbitrarily, sometimes it is defined by human ability to perform the same task. Take, for example, an experiment that is discussed in more detail in Part III, section 2.3: sarcasm detection. Assuming equal distribution of sarcastic and non-sarcastic phrases, statistical probability of guessing correctly is 50% because there are two possible outcomes (sarcastic or not sarcastic). Human ability to detect sarcasm is found to be much higher than guessing arbitrarily, so human performance makes for a more ambitious baseline.[35]

4 Machine Learning

This section provides a brief introduction as it relates specifically to Comp Ling tasks.[36] Generally speaking, the type of machine learning system utilized for Comp Ling consists of 3 parts: feature extraction, training, and testing. Once features of the kind discussed in Section 2 have been extracted, the data is divided into two sets: training and testing. The training set is customarily larger than the testing set. During the training phase, the system examines the features associated with the individual documents that make up the corpus and “learns” how to classify them. It builds a model using the given data. That model is then assessed using the testing set. The testing set should have data which is new to the system, so that the application is not merely reapplying the same classification as done in the training set. After all, the goal is to create an application which can predict the classification of never-before-seen data.

33 Also known as F1 score, F-measure, or balanced F-score.

34 (Mitkov, 2005)

35 (González-Ibáñez, Muresan, & Wacholder, 2011)

36 In fact, this paper will not even be distinguishing between data mining and machine learning, which, while discrete fields, greatly overlap.

The machine learning task typically relevant to NLP usage is classification. A POS tagger, for example, takes a word in a phrase and classifies it as a noun, adjective, verb, or so on. The categories into which the items are grouped are called “labels.”

The training set should be representative of the data that the application will be used to classify. If the data is too “noisy,” the model created may fall victim to a problem called “overfitting,” where unimportant features are given too much predictive significance. Imagine trying to group a set of students by natural hair color when they are all wearing hats that cover their hair. The labels may be “blond,” “brown,” “red,” and “black.” One might write down traits of each student, such as skin tone, eye color, and ancestry. If all the blond students also happen to have blue eyes, the system may learn to over-exaggerate the likelihood of a blue-eyed person to have blond hair, even though it is perfectly possible to have blue eyes with any other hair color, or blond hair with other eye colors.

4.1 Supervised Learning

Supervised learning is the most common approach used in NLP, but requires extensive annotated data. This must often be done manually. Supervised learning is based around the existence of a known set of classes, an example of each of which is contained in the corpus. In the training step, each document is labeled with one of the given classes. The system then builds a classification model, typically by finding the relative importance of the already-extracted features. Supervised learning algorithms include Support Vector Machines, Neural Networks, and Naïve Bayes classifiers.

Supervised learning systems are seen in (Pang et al., 2002), (Mullen & Collier, 2004), (Mishne, 2005), (Pang & Lee, 2005), (Généreux & Evans, 2006), (Agarwal et al., 2011), (González-Ibáñez et al., 2011) discussed in detail in Part III.

4.2 Unsupervised Learning

Unlike supervised training, unsupervised training does not require annotated examples. This means that these systems can be run on “raw' texts (though they still go through pre-processing and feature extraction steps). Unsupervised systems are useful when the defining characteristics of the sets to be classified are unknown. Data mining techniques can be used to “discover” the classes.

Despite the advantage of not needing annotated data for training, unsupervised learning is not as common for the types of applications discussed in this paper, but is significant to other NLP tasks, and even more so to many AI tasks.

4.3 Other types of learning

There are also machine learning applications which fit neither the supervised nor unsupervised category. One such approach mentioned in this paper is “semi-supervised” learning in (Davidov et al., 2010), (Tsur et al., 2010), and (Pang & Lee, 2005). Though this approach varies from project to project (just as supervised and unsupervised learning do), the generalization is that semi-supervised learning uses both labeled and unlabeled data in order to build a classifier. (Go & Huang, 2009) uses "distant supervision" which minimizes the amount of hand-annotated data required by supervised learning by instead using noisy but automatically-producible annotations.

5 Summary

In order for a Comp Ling application to function, it must be designed with the particular characteristics of the target data in mind. A source like Amazon has fewer features that distinguish it from traditional texts, but this is only addressing on the average case; the linguistic anomalies of the internet must still be accounted for and dealt with. Sources like LiveJournal and Twitter have their own unique linguistic—and non-linguistic—forms that must be considered.

Next, the textual data undergoes feature extraction, in which the data is characterized according to its features. This is a particularly important step, as choosing features is essential to the success of a Comp Ling application. Some basic and often-used features include unigrams, named entities, POSs, and POS patterns.

To judge the success of an experiment, recall, precision, and f-score measures can be used for certain types of experiments. By utilizing these measures, the experimental outcomes toward the same goal can be compared with regard to effectiveness, even if they test different features of the linguistic data.

Finally, most modern Comp Ling applications use some form of machine learning. Classification problems largely follow the same process: feature extraction, training, and testing. Types of machine learning include supervised, unsupervised, semi-supervised, and distant learning.

Part III
Sentiment Analysis

1 *Sentiment Analysis*

Sentiment analysis (**SA**)—also referred to as subjectivity analysis, review mining, or appraisal extraction[37]—is the task of creating a computer program that can automatically determine the emotive quality (the sentiment) of a piece of text. In the earliest incarnations of the task, the goal was to ascertain the polarity—positive or negative—of a text. This task has been more or less “mastered,” causing research to shift focus to more challenging, complex tasks such as identifying precise affective states (e.g., anger, happiness, depression) or detecting the presence of sarcasm. There are now a wide variety of extensions of applications of the original SA task.

SA can be compared to other types of text classification problems—it is essentially classifying texts among positive, negative, and sometimes neutral labels. The related task of opinion mining classifies text as opinionated or factual (without the component of characterizing the text's polarity). However, the two are unlike other classification tasks in that the text is subjective and opinionated, not objective and factual. Subjectivity is not the same as sentiment, nor is objectivity the same as neutral sentiment. Unlike “traditional” text classification, which may include numerous labels, SA includes only a handful of labels over many domains and authors.[38]

A typical opinionated piece of text can be considered to have several components: the opinion, the opinion target, and the opinion holder. The opinion is the sentiment that is analyzed by the system. The opinion target is that to which the opinion applied, e.g., the subject of a product review. The opinion holder is most typically the author of the opinionated text, though determining the opinion holder can sometimes be a challenge in and of itself.[39]

(Feldman, 2013) describes SA by the different “levels” at which it may be performed, each of which presents a slightly different task. The “biggest,” least-specific, and simplest form is document-level

37 (Pang & Lee, 2008)
38 (Pang & Lee, 2008)
39 (Pang & Lee, 2008)

SA, in which the overall opinion of the document's author is determined. SA can also be performed at the sentence level, where each sentence is judged first for objectivity or subjectivity, then further analysis only if the sentence is found to be subjective. This is a finer-grained approach which allows for multiple opinions in the same document, usually all about the same opinion target.

Document-level and sentence-level SA are fine when opinions are consistent at the relevant level of specificity. But what about a sentence which mixes sentiments, as in the following sentence?

The writing was fast-paced, but I didn't like the characters.

Much opinionated text not only expresses an overarching opinion, but gives reasoning toward that final opinion. This might be in terms of "pros" and "cons." In the example above, two traits of one opinion target are reviewed: the writing and the characters. (Lui, 2010) proposes a hierarchical approach to SA which involves identifying the main product, its components, and attributes of both the main product and its components. The term *feature* is used, but this is not to be confused with the linguistic features discussed above.

1.1 Building a Corpus, Revisited

Because of the nature of the task, SA is applied to pieces of text which contain subjective statements. SA's recent explosion of research is often linked to the increasing availability of opinionated text via the web.[40] Platforms such as Twitter, Livejournal, Amazon, movie-, restaurant-, and service-reviewing sites, and social media have given voice to consumers more than ever before. People also express their opinions about political issues and news events in ways never seen before.

1.2 Applications

With all the newly-available opinionated text on the web, SA can be used to help users navigate such texts. Product reviews constitute just a portion of the available opinionated text, yet there are myriad applications of SA to that portion alone, including automatically sorting reviews in terms of polarity and degree of polarity and summarizing the sentiment of reviews.

But the usefulness of SA is not limited to just individual users. Businesses are also quite interested in making use of SA. Along with the popularization of SA came the sudden boom in buzzwords such as "brand monitoring," "buzz monitoring," "online anthropology," "social media monitoring and analysis," "market influence analytics," "conversation mining," and "online consumer intelligence," all meant to indicate the use of SA to gauge public opinion with regard to a brand or product.[41] Companies are investing in "business intelligence" technologies to monitor social media that would otherwise swamp human abilities to keep up.

(Pang & Lee, 2008) and ((Lui, 2010) attribute an explosion of interest in SA research around 2001 partially to the sudden realization of the variety of applications in both academia and business. As recently as 2010, it was estimated that at least 20-30 companies offered SA services within the US alone.[42] SA is useful both as a core technology and as a subcomponent of other systems; SA could be used as a component for recommendation systems, detecting aggressive language, and predicting the rise of trends.[43]

40 (Lui, 2010; Pang & Lee, 2008)
41 (Pang & Lee, 2008)
42 (Lui, 2010)
43 (Pang & Lee, 2008)

2 *Specifics*

Now that the basics of SA have been laid out, the focus of this paper shifts to the role of linguistics in specific SA tasks. These tasks consist of binary polarity (positive-negative) classification, affective state identification, and sarcasm detection. Informally, these are organized in order of increasing complexity and difficulty; each has its individual challenges, but the less-complex tasks are typically more thoroughly researched and the more-complex tasks tend to use the less-complex ones as subcomponents. Each section will include an overview of the task, the challenges unique to the task, some approaches, applications, and further reading. The role of linguistics will be touched upon for each task.

2.1 Polarity / Opinion Scoring

Much of the work done in the field of sentiment analysis falls under the heading of binary classification of text as simply positive or negative.[44] “Neutral” is another common label, though this can also be used to describe objective text, which is slightly different from an unenthusiastic opinion. This task very naturally leads into classifying *degrees* of polarity, often in terms of units familiar to the reader as "stars," "points," and so on. This category of tasks is known by quite a few names, including "binary classification, " "sentiment polarity classification," "polarity classification," and "sentiment classification." [45] In this paper it will be referred to as "polarity classification."

Approaches to the task of polarity classification can be roughly grouped into two categories[46]: text classification and generation of a “sentiment lexicon.”

2.1.1 Text Classification Approach

(Pang et al., 2002) and (Pang & Lee, 2005) address the problems of sentiment polarity and degree, respectively, with approaches very similar to that of “classic” text classification. (Go & Huang, 2009)'s system also classifies real-time Twitter data pretty similarly to the generalized classification algorithm in Part II, Section 4.

2.1.2 Sentiment Lexicon Approach

The intuition behind generating a sentiment lexicon is that certain words will have inherent *semantic orientation*, or positive or negative value. For example, “despise” is an extremely negative word, so it is given “negative weight.” [47]

However, the sentiment lexicon approach quickly raises problems. For example, consider this sentence in a hypothetical movie review:

I usually despise Westerns, but this was an exception.

In the lexicon approach, the word "despise" may be very negatively weighted, and the other words in the sentence neutrally weighted, likely resulting in a negative rating on average. A human reader would probably say this is a neutral or possibly very positive review, largely dependent on the sentences following. Therefore polarity weight of “despise” should be flipped. However, there is also currently no

44 (Pang & Lee, 2008)
45 (Pang & Lee, 2008)
46 As always, with some outliers.
47 In the sense that it is given weight toward the negative side, not that weight is removed.

way to accurately capture negation. It is not a simple problem of looking for the words "no" or "not" in close proximity to the lexical items. In the example sentence above, neither of these words appears, yet between "usually" and “but”, "despise" is negated for this particular instance.

There are also words which typically have no polarity weight, but may have strong weight for a particular context. "Delicate" may be a positive word when describing an intricate glass sculpture, but is rarely positive when applied to a truck. Compare:

This forest is very lush and green.

The sushi looked kind of green.

Context is very important to the polarity weight of particular words.

Compilation of a sentiment lexicon can be done manually, but machine learning techniques have been shown more effective than humans at predicting what are called “discriminating” words.[48] Automatic generation of a sentiment lexicon is a more common method, as described in (Taboada, Brooke, Tofiloski, Voll, & Stede, 2011), sometimes referred to as “bootstrapping”: a small set of “seed words” with strong positive or negative connotations, and assume the surrounding words have similar (if weaker) polarity.

2.1.3 Linguistic features

Though many NLP tasks benefit from a term frequency feature, the initial unigram results of (Pang et al., 2002) suggests that term *presence* produces better performance than term frequency, for the task of predicting the sentiment of a movie review document. Further testing using term presence instead of frequency showed unigrams more effective than using bigrams, only terms tagged as adjectives, including the position of the word in the document, or combinations of unigrams and bigrams or unigrams and position. This highlights the contrast between “traditional” text classification, in which frequency has been shown more effective, and polarity classification. Naïve Bayes, Maximum Entropy, and Support Vector Machine algorithms were used, but showed relatively little variation in performance (though SVMs typically showed the greatest performance).

The SO-CAL system from (Taboada et al., 2011), which uses a lexicon-based approach, also makes use of POS features. The “seed words” are all adjectives, but nouns, verbs, and adverbs are also given positive and negative weight. Additionally, “amplifiers,” adjectives that modify the degree of other words, are given percentage values as to how they affect the polarity of the other words. A phrase like “truly awful” would be given a negative polarity by “awful,” amplified by the modifier “truly.”

48 (Kim, Li, & Lee, 2009; Pang, Lee, & Vaithyanathan, 2002)

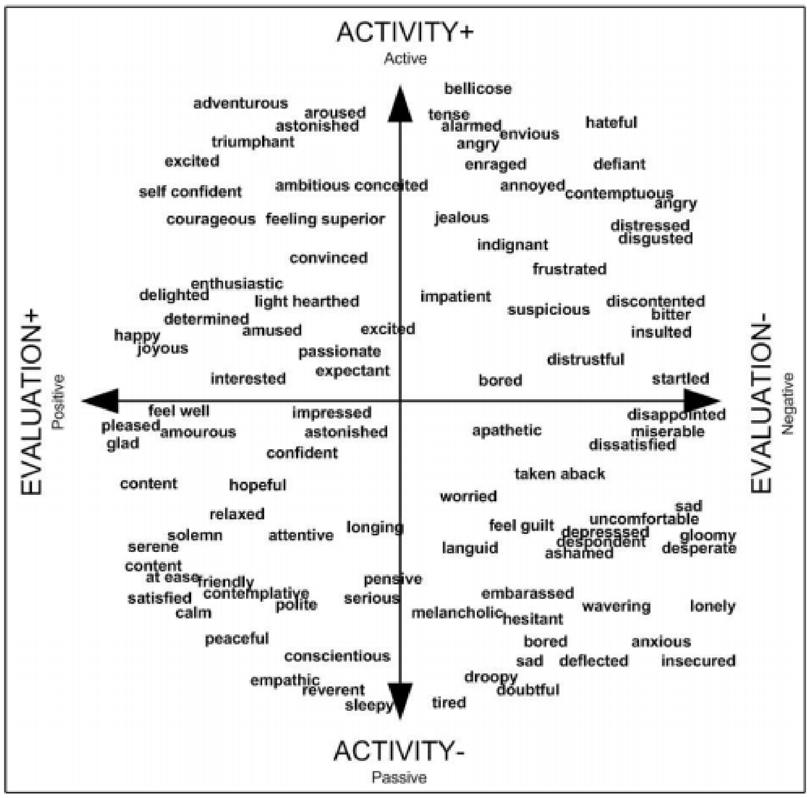


Figure 7: Representation of (Scherer et al., 2006)'s typology of affective states[49]

2.2 Affective State / Mood Classification

The second task to be examined is that of determining the affective state—that is, emotion—of a piece of text. Like polarity detection, this can be applied to product reviews; more frequently, however, affect discrimination is applied to blogs. Because Livejournal gives authors the option to tag their entries with emotions, it is a particularly popular source of data for the mood classification task. It saves some effort in annotating the entries, but also adds some complications.

2.2.1 Challenges

Finding features which are consistent among many different authors/styles for particular emotions is a particular problem for the mood classification task. Though the "same" emotion may be felt by multiple people, it may be expressed uniquely by each one. In the face of this fact, human ability to identify the emotional state of others is quite impressive. Unfortunately, appreciation for humans' ability to do so does not make the task any easier to simulate in computers. The corpus must be very carefully managed to ensure that the task of mood classification does not become one of authorship attribution—a related but separate task in which stylistic features are analyzed in order to identify the author of an otherwise anonymously-written text.

49 (Généreux & Evans, 2006)

Furthermore, annotations may be mis-annotated by the authors marking an emotion unrelated to the text of the blog entry. For example, the author may just happen to feel tired while writing a purely informational post. It is worth noting that, unlike the other tasks, mood classification is not performed only on subjective sentences; the sentence “I am happy today,” should be quite useful to a mood classification system, though it arguably resembles a factual statement.

Finally, in addition to providing 132 “preset” moods, Livejournal also allows authors to type in their own emotional state markers, creating a number of emotions which only appear once in the dataset. These are typically discarded. In fact, (Mishne, 2005) uses only the 40 most frequently-marked moods in order to reduce the number of classes and hopefully reduce variance in the data set.

2.2.2 Approaches

Again, for the task of determining affective states, one approach is to build a type of sentiment lexicon. This is the approach utilized by (Mishne, 2005). Of course, all the problems of the simpler polarity sentiment lexicon apply, exponentially to the degree of the number of distinct emotions identified. The problems of negation, context, and so on apply to each emotion, in addition to the relationships between different emotions.

Another approach builds on (Scherer, Dan, & Flykt, 2006)'s typology of affective states and their relationships, which maps them onto a two-dimensional plane. One of these dimensions corresponds to polarity. The other corresponds to activity (active or passive). This model was used in (Généreux & Evans, 2006)'s experiments on affective states in web logs. Though the results were a less than impressive, the authors concluded that (Scherer et al., 2006)'s model is useful as a basis for locating affective states.

2.2.3 Linguistic features

(Mishne, 2005) makes use of what are described as “classic” text analysis features, including term frequency counts for unigrams; frequencies of POSs; frequencies of word lemmas; length of blog posts in terms of bytes and number of words; and average sentence length in bytes and number of words. Measures of semantic orientation from two sentiment lexicons were used, in addition to a statistical feature called Pointwise Mutual Information (**PMI**) which measures the degree of association between two terms. Specifically, the degree of association between a particular word and a particular mood was measured, including that between “homework” and “annoyed”; “nap” and “great”; and “goodnight” and “sleepy.” Finally, knowledge of the medium—as detailed in Part II, section 1.1.2—was used as an advantage, by using frequency of emphasized words and special symbols as features.[50]

Because (Généreux & Evans, 2006) is focused on testing the practicality of (Scherer et al., 2006)'s typology, only basic, previously-tested features (unigrams, POS, etc) are used. This allows for experimentation with the geometric distance between moods as laid out in the typology without interference from unpredictable features.

Neither (Mishne, 2005) nor (Généreux & Evans, 2006) proved particularly successful at classifying moods. Both classified for active/passive and positive/negative binaries, but neither obtains better than 76% accuracy, calculated comparing the application's classification predictions to the moods annotated by the blog authors. Though this is well above a 50% arbitrary positive-negative guess baseline, many of (Mishne, 2005)'s results for particular moods hovered around the 50-60% accuracy mark. The subjective

50 Word emphasis rendered in ALL CAPS, *with asterisks* or _with understores_ are common when bold and italic formatting are not provided. Special symbols included two classes: punctuation and emoticons.

nature of defining moods still eludes current research, but is likely to be a rich area of research into the future.

Notice that, while the features used include basic ones such as unigrams and POSs, they also include more advanced features like PMI measures. Additionally, a psychological typology is pulled into play and deemed useful for future work, though the initial results are not dazzling. These features are more linguistically and psychologically informed than those used in the simpler polarity task.

2.2.4 Applications

The ability to determine the emotion behind a piece of text has quite a variety of applications, from assisting behavioral scientists, to improving doctor-patient interaction, to filtering search by mood, to identifying cohesive communities.[51] Almost any area that involves interaction can be improved by increasing the apparent "empathy" of the computer system.

2.3 Sarcasm Detection

The final task to be examined in this paper is that of sarcasm detection. Sarcasm is tricky to define, to say the least; broadly, one might consider sarcasm to be the act of saying the opposite of what is actually meant, flouting the Gricean maxims, including those of quality and manner. However, the implicit nature of sarcasm makes it difficult for even humans to detect, never mind for humans to tell computers how to detect. (Davidov et al., 2010) expands on facets of sarcasm, including:

- reference to unmentioned context (termed "use of universal knowledge"), as in: *She is as graceful as ever*, where the reader must know that "she" has never been particularly graceful (or has recently had an incident of clumsiness).
- combination of multiple sarcastic phrases, as in: *I just love our neighbors' noisy barbecues. It's so considerate of them to neglect to invite us every time. The music until 2am on weeknights is really the cherry on top, though.*
- embellishments or over-exaggeration, as in: *Wow, love the band's new pop sound.* This is a difficult aspect to distinguish from sincere enthusiasm on its own.

Despite knowledge of these facets of sarcasm, each of these components is troublesome to directly build into a model.

That sarcasm is difficult for even other humans to reliably detect makes it a particularly interesting problem to which to apply techniques from data mining.

Like mood classification, getting proper annotation for a sarcastic corpus presents many challenges. (Davidov et al., 2010), (Tsur et al., 2010), and (González-Ibáñez et al., 2011) all relied on self-annotation by looking for the #sarcasm hashtag on tweets. This relies on Twitter's millions of users to consistently tag their tweets—unlikely. Though this technique is a very noisy, unreliable way to collect sarcastic data, it is one of very few practical options for building the necessary corpus.

2.3.1 Linguistic features

(Davidov et al., 2010) and (Tsur et al., 2010)'s features are centered around classifying words into "high-frequency" and "content" groups, and then extracting patterns of those, very similar to the POS pattern extraction seen in Part II, Section 2.4. Sentence length, counts of different types of punctuation possibly indicative of sarcasm, and counts of capitalized/all caps words per sentence are also utilized as

51 (Mishne, 2005)

features. These features were tested on both Twitter and Amazon datasets, but were found to be more successful on Twitter; the improvement is attributed to Twitter's context-free nature, such that the sarcasm must be made more obvious.

(González-Ibáñez et al., 2011)'s approach is based more in lexical and pragmatic features such as unigrams, dictionaries, emoticons, and reply targeting.[52] The highest accuracy achieved in this experiment was 71%, but even individual humans were only able to score as high as 82.95%.

Like mood classification, these features used in these sarcasm detection experiments are more complex than simple unigrams and POSs, involving even classification of words for (essentially) content and function words.

2.3.2 Applications

Though most users seem to realize that sarcasm just does not always translate in textual form, the interactive nature of the internet makes its appearance almost inevitable, and often in inappropriate places. For example, highlighting the review excerpt, "Now my kids can learn how to be a right-wing president that utilizes illegal and inhumane drones to kill innocent children in Pakistan!" as the top comment on a toy military drone was probably not intentional on Amazon's part.[53] Sarcasm detection could be used to improve many NLP applications, including review summarizations such as that one, dialogue systems, and review ranking systems.[54]

3 Summary

SA in general can be summarized as automatically detecting and or characterizing the sentiment of a piece of subjective text. In its most low-level form, this means merely classifying the text's sentiment polarity (positive or negative). This extends naturally into the task of determining the *degree* of polarity, as in "mildly positive" and "strongly negative," typically in terms of discrete units such as points or stars.

Other branches of SA include affective state classification and sarcasm detection. The former goes beyond simple polarity to classify for specific moods: "angry," "happy," "sleepy," and so on. A typical way to build upon the work done for polarity is to add an active/passive dimension, and "plot" emotions according to these two axes.

The latter, sarcasm detection, is possibly among the most intriguing type of SA task, because even humans have not completely mastered distinguishing sarcasm from sincerity. Naturally, despite enthusiastic attention to the problem, results for this task as yet remain somewhat unimpressive.

As tasks increase in complexity, the features extracted from the data become increasingly abstract, and either rely upon or parallel models from psychology and linguistics. The simplest tasks of polarity detection and even polarity scoring use simple features such as unigrams and POSs. Sentiment lexicons are built by using connotations and associations. Approaches to mood classification add an extra dimension which relies upon outside psychological models. Sarcasm detection approaches utilize concepts such as "content" and "non-content" words, which exist in parallel in linguistic theory. Definitions for sarcasm are also based on linguistic models including Grice's famous maxims.

52 Recall that *@OtherUser* is referred to as the target.

53 (Drewboy64, 2013) The author is not certain that such an application would make the world a better place.

54 (Davidov et al., 2010)

Part IV
Conclusions

Sarcasm detection, mood classification, polarity classification—these are just a few of the tasks under the umbrella of sentiment analysis, itself just a tiny portion of the field of computational linguistics. Yet even these three tasks open up worlds of possibility, especially with regard to human-computer interaction. Imagine product reviews synthesized from thousands of individual reviews, constructed to highlight the often-mentioned pros and cons of the item of your interest. Imagine your computer sensing that now might be a good time to turn on “easy mode”—that edge in your voice suggests that right now is probably not the proper time for complicated menus and commands. Imagine that Siri understands that you did not *really* want to find the closest “vegetarian butcher.” Gimmicky uses of technology? Maybe. But the same technology can be utilized in subtle and more practical ways, too.

This paper has given an introduction to computational linguistics, with special focus on just one of its many subfields. The reader has been familiarized with the importance of tailoring a corpus, been exposed to some common linguistic features, models, and tools, and been given a foundation for understanding the outline of a machine learning system. These fundamentals can be applied to more than just the sentiment analysis task that followed them in this paper—they have been generalized to be adaptable to as many computational linguistics tasks as possible for one work.

The combination of linguistic data available via the internet and advancements in AI techniques has changed the game when it comes to computational linguistics; its practitioners no longer need focus on hand-writing tedious linguistic rules. Instead, given enough data, systems can be created to “learn” the rules of language, almost like a child learning the language around it through example. However, even such seemingly-miraculous learning systems cannot be created without intuitions about the data. It is essential not only to know the properties of the data operated on, but also to come to the task with intuitions what features may be used to solve it.

By highlighting the features of the most complex sentiment analysis tasks, this paper has attempted to show that further research can and should benefit from the insight of linguists. That is where linguists enter the field as it is today—not by trying to outline the totality of linguistic fact, but by providing insight into properties of language worth investigating.

References

Agarwal, A., Xie, B., Vovsha, I., Rambow, O., & Passonneau, R. (2011). Sentiment Analysis of Twitter Data. In *Proceedings of the Workshop on Language in Social Media (LSM 2011)* (pp. 30–38). Portland, Oregon: Association for Computational Linguistics.

amazon.com. (2013). Invisible Cities: Italo Calvino. http://www.amazon.com.

Bird, S., Loper, E., & Klein, E. (2009). Natural Language Processing with Python. O'Reilly Media Inc.

Calvino, I. (1978). *Invisible Cities*. (W. Weaver, Trans.). Houghton Mifflin Harcourt.

Crystal, D. (2011). *Internet Linguistics: A Student Guide*. Routledge.

Danescu-Niculescu-Mizil, C., Kossinets, G., Kleinberg, J., & Lee, L. (2009). How Opinions are Received by Online Communities: A Case Study on Amazon.com Helpfulness Votes. In *International World Wide Web Conference Committee*. Madrid, Spain: Association for Computing Machinery.

Davidov, D., Tsur, O., & Rappoport, A. (2010). Semi-Supervised Recognition of Sarcastic Sentences in Twitter and Amazon. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning* (pp. 107–116). Uppsala, Sweden: Association for Computational Linguistics.

Drewboy64. (2013). *Maisto Fresh Metal Tailwinds 1:97 Scale Die Cast United States Military Aircraft - US Air Force Medium Altitude, Long Endurance, Unmanned Aerial Vehicle*. http://www.amazon.com.

Elpeninsular. (2013). Twitter / elpeninsular: @albertochimal RT @Radiolab ... *Twitter*.

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

Fritz, K. (2013). Twitter / ladykatefritz: @Radiolab @LievSchreiber ... *Twitter*.

Généreux, M., & Evans, R. (2006). Distinguishing affective states in weblog posts. In *AAAI Spring Symposium on Computational Approaches to Analyzing Weblogs*. Stanford University, California: American Association for Artificial Intelligence.

Go, A., & Huang, L. (2009). Twitter Sentiment Classification using Distant Supervision. Technical report. Stanford.

González-Ibáñez, R., Muresan, S., & Wacholder, N. (2011). Identifying Sarcasm in Twitter: A Closer Look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics:* (pp. 581–586). Portland, Oregon: Association for Computational Linguistics.

Jenkins, M.-C. (2009). How Sentiment Analysis works. Slide presentation.

Jurafsky, D. (n.d.). Sentiment Analysis. Stanford University Natural Language Processing. Educational presentation.

Kim, J., Li, J., & Lee, J.-H. (2009). Discovering the Discriminative Views: Measuring Term Weights for Sentiment Analysis. In *Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP* (pp. 253–261). Suntec, Singapore: ACL and AFNLP.

Klein, D., & Manning, C. D. (2003). Accurate Unlexicalized Parsing. In *Proceedings of the 41st Meeting of the Association for Computational Linguistics* (pp. 423–430).

Lui, B. (2010). Sentiment Analysis and Subjectivity. In N. Indurkhya & F. J. Damerau (Eds.), *Handbook of*

Natural Language Processing, Second Edition.

Manning, C. D., Raghavan, P., & Schütze, H. (2009). Introduction to Information Retrieval. *Cambridge University Press*.

Marcus, M., Taylor, A., MacIntyre, R., Bies, A., Cooper, C., Ferguson, M., & Littman, A. (1999). The Penn Treebank Project.

Mishne, G. (2005). Experiments with Mood Classification in Blog Posts. In *Style2005 - the 1st Workshop on Stylistic Analysis Of Text For Information Access*.

Mitkov, R. (2005). *The Oxford Handbook of Computational Linguistics*. (R. Mitkov, Ed.). Oxford University Press.

Mullen, T., & Collier, N. (2004). Sentiment analysis using support vector machines with diverse information sources. In *In Proceedings of Conference on Empirical Methods in Natural Language Processing*.

Pang, B., & Lee, L. (2005). Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the ACL* (pp. 115–124). Ann Arbor, Michigan: Association for Computational Linguistics.

Pang, B., & Lee, L. (2008). *Opinion Mining and Sentiment Analysis*. *Foundations and Trends® in Information Retrieval* (Vol. 2). Foundations and Trends(R) in Information Retrieval.

Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up? Sentiment Classification using Machine Learning Techniques. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Plebani, D. (2011). Twitter / DamianoPlebani: I'd have liked... *Twitter*.

Porter, M. (2006). Porter Stemming Algorithm.

Radiolab. (2013). Twitter / Radiolab: What if the moon were just ... *Twitter*.

Scherer, K. R., Dan, E. S., & Flykt, A. (2006). What determines a feeling's position in affective space? A case for appraisal. *Cognition and Emotion*, *20*(1), 92–113.

Taboada, M., Brooke, J., Tofiloski, M., Voll, K., & Stede, M. (2011). Lexicon-Based Methods for Sentiment Analysis. *Computational Linguistics*, *37*(2), 265–307.

Tsur, O., Rappoport, A., & Davidov, D. (2010). ICWSM – A Great Catchy Name: Semi-Supervised Recognition of Sarcastic Sentences in Online Product Reviews. In *International AAAI Conference on Weblogs and Social Media* (pp. 162–169). Association for the Advancement of Artificial Intelligence.

Wang, H., Can, D., Kazemzadeh, A., Bar, F., & Narayanan, S. (2012). A System for Real-time Twitter Sentiment Analysis of 2012 U.S. Presidential Election Cycle. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics* (pp. 115–120). Jeju, Republic of Korea: Association for Computational Linguistics.

Selected References by Topic

Polarity / Opinion Scoring

Go, A., & Huang, L. (2009). Twitter Sentiment Classification using Distant Supervision.

Kim, J., Li, J., & Lee, J.-H. (2009). Discovering the Discriminative Views: Measuring Term Weights for Sentiment Analysis. In *Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP* (pp. 253–261). Suntec, Singapore: ACL and AFNLP.

Pang, B., & Lee, L. (2005). Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the ACL* (pp. 115–124). Ann Arbor, Michigan: Association for Computational Linguistics.

Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up? Sentiment Classification using Machine Learning Techniques. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Taboada, M., Brooke, J., Tofiloski, M., Voll, K., & Stede, M. (2011). Lexicon-Based Methods for Sentiment Analysis. *Computational Linguistics*, *37*(2), 265–307.

Wang, H., Can, D., Kazemzadeh, A., Bar, F., & Narayanan, S. (2012). A System for Real-time Twitter Sentiment Analysis of 2012 U.S. Presidential Election Cycle. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics* (pp. 115–120). Jeju, Republic of Korea: Association for Computational Linguistics.

Affective State / Mood Classification

Généreux, M., & Evans, R. (2006). Distinguishing affective states in weblog posts. In *AAAI Spring Symposium on Computational Approaches to Analyzing Weblogs*. Stanford University, California: American Association for Artificial Intelligence.

Mishne, G. (2005). Experiments with Mood Classification in Blog Posts. In *Style2005 - the 1st Workshop on Stylistic Analysis Of Text For Information Access*.

Scherer, K. R., Dan, E. S., & Flykt, A. (2006). What determines a feeling's position in affective space? A case for appraisal. *Cognition and Emotion*, *20*(1), 92–113.

Sarcasm Detection

Davidov, D., Tsur, O., & Rappoport, A. (2010). Semi-Supervised Recognition of Sarcastic Sentences in Twitter and Amazon. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning* (pp. 107–116). Uppsala, Sweden: Association for Computational Linguistics.

González-Ibáñez, R., Muresan, S., & Wacholder, N. (2011). Identifying Sarcasm in Twitter: A Closer Look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics:* (pp. 581–586). Portland, Oregon: Association for Computational Linguistics.

Tsur, O., Rappoport, A., & Davidov, D. (2010). ICWSM – A Great Catchy Name: Semi-Supervised Recognition of Sarcastic Sentences in Online Product Reviews. In *International AAAI Conference on Weblogs and Social Media* (pp. 162–169). Association for the Advancement of Artificial Intelligence.

Appendix I: Abbreviation Glossary

AI	Artificial Intelligence
BoW	Bag of Words
Comp Ling	Computational Linguistics
IR	Information Retrieval
IE	Information Extraction
NLP	Natural Language Processing
PMI	Pointwise Mutual Information
POS	Part-of-speech
SA	Sentiment Analysis
SGML	Standard Generalized Markup Language