SEMI-SUPERVISED SUBJECTIVITY CLASSIFICATION AND

APPLICATION TO JARGON HEAVY CORPORA

by

Jason Michael Switzer

APPROVED BY SUPERVISORY COMMITTEE:

_____

Dr. Latifur Khan, Chair

_____

Dr. Vincent Ng

_____

Dr. Weili Wu

SEMI-SUPERVISED SUBJECTIVITY CLASSIFICATION AND APPLICATION TO JARGON HEAVY CORPORA

by

JASON MICHAEL SWITZER, B.S.

THESIS

Presented to the Faculty of

The University of Texas at Dallas

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

THE UNIVERSITY OF TEXAS AT DALLAS

December 2010

ACKNOWLEDGEMENTS

I would like to thank everyone who has helped me in my journey. I would like to thank God for giving me the ability, passion, and opportunity to achieve. I would like to thank all of my instructors, professors and classmates who have pushed, taught, and assisted me when it was needed most. I would like to thank my family who has believed in me over the years. Lastly, I would like to thank my wife for having patience, understanding, love, and encouragement these many years. Without her, I would have been lost.

December 2010

SEMI-SUPERVISED SUBJECTIVITY CLASSIFICATION AND

APPLICATION TO JARGON HEAVY CORPORA

Publication No. _____

Jason Michael Switzer, M.S.
The University of Texas at Dallas, 2010

Supervising Professor: Dr. Latifur Khan

Semantic analysis of corpora containing heavy usage of jargon words and phrases introduces problems not commonly addressed by Natural Language Processing methods. Modern semantic analysis relies on data from unedited websites or other expertly written sources, which lack similar usage of jargon words and phrases. This thesis presents a system of semi-supervised lexicon learning algorithms that collate several manually labeled and clustered data sources, such as thesauri. In addition, this thesis demonstrates an improvement in performance of these pipeline classifiers by applying a boosting method. This paper presents a method of automatic shaping factor classification based on the most relevant words from a subjectivity lexicon.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATIONS

CHAPTER 1

INTRODUCTION

An impetus behind causal analysis of the Aviation Safety Reporting System, or similar corpora, involves identifying the underlying cause of the report in spite of the existence of jargon. An automatic approach to locating the words or clauses in an individual report (or sentence) is the most desirable method of causal analysis.

There has been an abundance of recent research focused in sentiment analysis. Sentiment analysis is an area of focus in the field of Natural Language Processing (NLP). Sentiment analysis, or subjectivity classification, involves classifying a word or phrase as positive or negative in most contexts. Subjectivity classification has applications to problems such as market analysis, public opinion, and customer feedback [1].

The Aviation Safety Reporting System corpus, ASRS, contains voluntarily submitted aviation incident reports. NASA builds the ASRS corpus from incidents reported by pilots, air traffic controllers, dispatchers, flight attendants, and mechanics. NASA collects approximately 5,000 reports every month from everyone involved in the aviation system [38]. Given the source of the reports, the terminology frequently used in each report often contains jargon words or phrases specific to the aviation system. The high occurrence of jargon presents issues not present with other corpora, such as the New York Times Annotated Corpus [20], a heavily edited and highly regarded corpus compiled from the New York Times newspaper.

The reports in the ASRS corpus typically refer to deficiencies in the aviation system. As such, there should be a number of words and phrases used in negative context. The subjective words in the reports either are direct causes or surround the root cause of an incident.

The primary motivation of this research is to provide a system for automatic classification of the shaping factors that determine the cause of an incident. This research will demonstrate that a subjectivity lexicon is capable of automatically identifying many words and phrases indicative of the shaping factors.

1.1 The Approach

The first task required to analyze the ASRS corpus is to build a large and highly accurate lexicon of subjective words and phrases. Presented here is a framework to allow an arbitrary number of subjectivity classification algorithms connected by passing their resulting subjectivity lexicon hypothesis to subsequent algorithms. Algorithms communicate by passing their subjectively classified lexicon between phases of execution. This protocol is simple enough to allow future tools to extend upon and utilize the subjectivity lexicon.

A set of Pipelined Subjectivity Classifiers (PSC) determine the subjectivity classification for a set of words and phrases. Each PSC operates independently of every other PSC, either semi-supervised or fully supervised, to classify and discover subjective words. For example the Moby Seed Lexicon uses the manually labeled synonym sets (or synonym clusters), whereas the WordNet algorithm explores WordNet synsets to discover and label instances. All semi-supervised learning algorithms take advantage of the unified lexicon format and classify the existing and new instances based on the output from a prior PSC. This pipeline concept is visualized as a directed graph of PSC algorithms.

Each of the PSC algorithms will operate on one or more data sources and will build upon the lexicon of prior PSC algorithms. For that reason, differences of opinions between PSC algorithms may result in the removal or complete reversal of a polarity score. For example, the Moby Seed Lexicon algorithm may assign a word a positive score of 1 and pass this to the WordNet algorithm. At that point, the WordNet algorithm may discover enough negative associations to reassign the word's score to -1, changing the polarity from positive to negative. Once a PSC algorithm builds a subjectivity lexicon, the performance is improved by applying a boosting algorithm.

This thesis presents the results of a large number of experiments to determine the best performing lexicon. This definitive lexicon, referred to as the ASRS Reference Lexicon (ARL), will assist in determining a set of words that most embody the shaping factors. The shaping factors, provided by Posse et al. [34] are useful training features to other classifiers. When a word strongly correlates to a shaping factor, it will have a high information gain. The words from the ARL that have the highest information gain are presented as the most indicative of the respective shaping factor.

1.2 Experimental Context

The related research has shown that thesaurus based classification can achieve a higher level of accuracy when compared to the authoritative General Inquirer lexicon [1]. When mixed with other semi-supervised learning methods, this technique shows certain combinations have favorable results.

To test the semi-supervised subjectivity classifiers, each possible combination will build a distinct subjectivity lexicon. These lexicon files are then compared to the General Inquirer and

our manually created lexicon for accuracy. The result is 37 lexicons are created; once without boosting, once with AdaBoost applied for 3 iterations, and once with InvBoost applied for 3 iterations. A total of 222 lexicons created and checked for accuracy against two reference lexicons. Afterwards, the ASRS corpus is searched for the words in the ARL and the words with the highest information gain for each shaping factor is reported.

1.3 Contributions

This thesis explores a variety semi-supervised learning algorithms for subjectivity classification. The main contribution lies with the pipelined combinations of the semi-supervised learning algorithms. This research also demonstrates the application of AdaBoost and a modified AdaBoost, known as InvBoost, as a stage of post processing to improve the performance of the individual PSC algorithms. Lastly, this research will demonstrate an automatic method of using the subjectivity lexicon to identify the words most frequently associated to the ASRS shaping factors.

1.4 Thesis Outline

The rest of the thesis is organized in the following manner. Chapter 2 outlines the prior work that forms the foundation of this thesis, emphasizing the concept of subjectivity classification, jargon word identification, and the ASRS corpus. Chapter 3 covers the background information for AdaBoost and Information Gain. Chapter 4 discusses the variety of training data sources used by the various PSC algorithms. Chapter 5 introduces the implementation details of the PSC algorithms and discusses their strengths and weaknesses. Chapter 6 interprets the results from the PSC algorithm combination experiments. Chapter 7

explores the complications involving semantic analysis of the ASRS corpus and application of the subjectivity lexicons. Finally, Chapter 8 summarizes the information presented here and discusses future work, including alternative data sources and PSC algorithms.

CHAPTER 2

RELATED WORK

Subjectivity classification is an area of research under the umbrella of Natural Language Processing that has garnered attention recently. This thesis draws on many ideas to accomplish the analysis of subjectivity classification towards jargon heavy corpora. This chapter discusses related research in the area of subjectivity classification.

The primary basis of this research is a paper by Mohammad et al. to develop an automatic classification scheme based on the Macquarie Thesaurus [1]. They developed a set of affix seed patterns [1] to mark antonymous word pairs and trained a subjectivity lexicon based on those 11 patterns. They also proposed a means to expand the subjectivity lexicon by using thesaurus entries for discovery and classification. They showed that this method was highly effective when compared against other methods such as Turney and Littman lexicon (TLL) and SentiWordNet. While the performance reported in the paper is impressive, this thesis will demonstrate even further improvements, both in the size of the trained lexicon and the accuracy.

Esuli and Sebastiani developed a method to assign semantic orientation labels to WordNet synsets, known as SentiWordNet [2]. Their technique augments WordNet by training a random walking semi-supervised classifier [5]. While this approach is similar to certain aspects presented here, it has deficiencies that this paper aims to rectify. Their random walk phase uses only the gloss information for a word, whereas WordNet 3.0 also provides synonyms that indicate a more direct relationship. In addition, their approach is heavily dependent on a single

data source, where this paper presents a method of combining multiple data sources and classifiers in the pipeline classification system. The Oxford English Dictionary is growing at a pace that WordNet simply cannot match [37]. For this reason, alternative methods, such as those employed in this thesis, must be incorporated to better accommodate an evolving language.

The secondary focus of this research is the work of Abedin et al. towards causal analysis of the Aviation Safety Reporting System [9]. They proposed using the Basilisk framework and a set of shaping factors to locate specific causes of incidents within each report of the corpus. Abedin provided the preprocessed ASRS corpus and the ASRS shaping factors training data. Mandala, Takenobu, and Tanaka [8] inspired the idea for traversing WordNet. Their research aimed at improving information retrieval by combining WordNet synset words with thesauri information. They utilized the SMART stop word list [25], of which the stop words used by this thesis are a subset [26].

Jesus Jane, Stephen Helmreich, and David Farwell presented an approach to automatically identifying jargon words in texts [36]. Their method was centered around lexical differences of two polarizing corpora, each corpora pair advocating opposing viewpoints. Their method targets text categorization based on decision tree learning models. This approach targets highly specific corpora (two opposing viewpoints), thus is not a general-purpose technique that will work on a corpus that lack a similar structure.

Hassan and Radev present a novel approach to randomly walking word relatedness graphs [5]. The polarity score of each word is determined from the average time spent traversing the graph until a labeled instance is found. The polarity score mechanism presented in this thesis is similar in nature because it represents the magnitude of positivity or negativity. That is, the

higher the score, or the greater the distance from 0, the more frequently it is determined to be positive. Their research suffers from the same limitations as Esuli and Sebastiani due to the dependency on WordNet.

Suzuki, Takamura, and Okumura developed a system of bootstrapping a semi-supervised classification method, Naïve Bayes, around contextual clues within the text [4]. For example, "Good, since the storage capacity of the laptop is high." demonstrates that "capacity is high" can be determined to be subjectivity positive from the contextual clue "Good". They did not cover co-training, or the use of independent learning methods, to train a subjectivity lexicon. Their research is limited to a single classifier, whereas the pipeline classification system presented here is a collection of classifiers.

Wilson Wiebe, and Hoffman developed the Multi-Perspective Question Answering subjectivity lexicon [6] that becomes a highly valuable seeding lexicon for the pipeline classification system presented by this thesis.

Sokolova, Japkowicz, and Szpakowicz provided a survey of evaluation metrics, such as the F-measure and discriminate power [7]. The pipeline classification system can report most of the methods provided by this paper, but only the accuracy and F-1 scores will be used for comparison in Chapter 6.

CHAPTER 3

BACKGROUND

3.1 AdaBoost

Boosting is a machine learning method that falls under the category of ensemble learning. Ensemble learning is a means of selecting a set of best performing hypotheses from a larger hypothesis space and using this ensemble to determine the best classification.

Boosting is a widely used method that emphasizes training instances for poor performing classifiers to improve their classification. To do this, the training data is weighted and the final classification is the weighted majority classification of every hypothesis generated during training. The output classification function selects the hypothesis that performs the best.

AdaBoost is a widely studied and accepted boosting algorithm that is straightforward and is applicable to any number of weak classifiers (classifiers that perform slightly better than random guessing). Figure 1 shows the algorithm structure [21].

AdaBoost.M1

1. Initialize the weights

$$w_i = \frac{1}{N}, i = 1, 2, \ldots N$$

2. For m = 1 to M
   a. Train classifier $h_i(x)$
   b. Compute the error

$$err_m = \frac{\sum_{i=1}^{N} w_i * I(y_i \neq h_i(x_i))}{\sum_{i=1}^{N} w_i}$$

   c. Compute α

$$\alpha_t = \log \frac{1 - err_m}{err_m}$$

   d. Update the weight

$$w_i = w_i * e^{\alpha_m * I(y_i \neq h_i(x_i))}, i = 1, 2, \ldots N$$

3. Output $H(x)$ classification

$$H(x) = sign\left[\sum_{t=1}^{T} \alpha_m h_i(x_i)\right]$$

where $I(y_i \neq h_i(x_i)) = \begin{cases} -1, & if\ y_i\ is\ correctly\ classified \\ 1, & if\ y_i\ is\ misclassified \end{cases}$

Figure 1. AdaBoost.M1 Algorithm

The AdaBoost algorithm needs a set of labeled training instances *S* of size *N* and a weak classifier *C*, to operate upon. The number of iterations, *M*, can either be predetermined or reached after the change in improvements has fallen below a threshold. AdaBoost will continue boosting until reaching the desired *M*, as seen in step (2). The computed error during step (2.b) is the normalized sum of the weights of misclassifications. The weight that is re-computed during step (2.d) will decrease for misclassifications and increase for classifications.

When the classifier, *C*, is a weak classifier performing slightly better than random guessing, then AdaBoost will return a hypothesis that classifies the training data with no error with a sufficiently large *M*.

The performance of the AdaBoost algorithm surprised researchers initially because it directly contradicts Occam's razor by improving classification as the hypothesis space gets more complex. It has been shown that the performance of AdaBoost improves as hypotheses are added because it approximates Bayesian learning methods [23, 32]. However, a down-side of boosting is that it can often reduce performance when trained with noisy data.

Many of the subjectivity classifiers presented by this thesis are poor performing weak classifiers. Boosting those weak classifiers early in the pipeline execution will improve the performance of future classifiers later in the pipeline. Section 5.4 presents boosting applied to the task of subjectivity classification, including a modification that allows for the direct use of the training weights for classification. Section 6.4 discusses the results of boosting the subjectivity classifiers.

3.2 Information Gain

Information gain becomes a vital role in this project once we have trained a large polarity lexicon proven accurate against the baseline lexicon. In order to understand why this is a valuable statistical property to report, a brief background in entropy and information gain is necessary.

The primary purpose of this statistical property, *information gain*, is to measure how well a feature or attribute divides the training and testing data points during classification. In order to

understand information gain, we must first explore another statistical property called *entropy*. Entropy is defined as a measure of impurity of a given set of data points.

The classical definition of entropy is the least amount information necessary to encode the classification. This definition is taken from information theory but has utility in many machine learning algorithms, such as the ID3 decision tree training algorithm [27]. As understood by other machine learning algorithms, including ID3, entropy indicates whether all samples from the collection of data points are have the same classification label. Informally, entropy measures the expectation of the training data.

Given a collection of data points, *S*, and a binary classification system, we will formally define entropy as follows [21]:

$$Entropy(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus} \quad (1)$$

where $p_{\oplus}$is the probability of positive data points in *S* and $p_{\ominus}$ is the probability of negative data points within *S*, and it is generally accepted that $0 \log_2 0 = 0$. For example, if there are 10 data points in *S*, 4 positive, and 6 negative, then $p_{\oplus}$will be 0.4 and $p_{\ominus}$will be 0.6. The resulting entropy will be

$$\begin{aligned} Entropy(S) &\equiv -0.4 \log_2 0.4 - 0.6 \log_2 0.6 \\ &\equiv 0.52877 + .44218 \\ &\equiv 0.97095 \end{aligned}$$

As the entropy approaches 1, there is a larger disparity, or impurity, amongst the data points in *S*. Conversely, as the data set becomes homogenous, the entropy will approach 0. To demonstrate this further, Figure 2, taken verbatim from [12], shows the entropy values that vary with the portion of positive $p_{\oplus}$ data points in *S*.

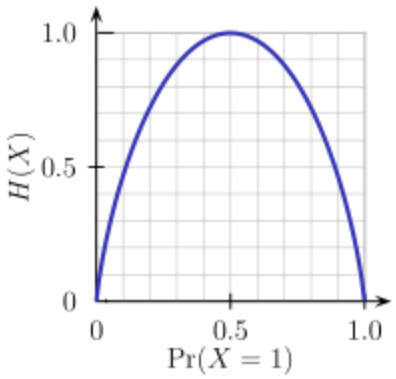

Figure 2. Boolean Entropy of $P(x_i = 1)$

Entropy is not limited to binary classification tasks. To calculate entropy for an arbitrary number of classification labels, we can make a minor change to Equation (1). From (1), we derive [21]:

$$Entropy(S) \equiv \sum_{i=1}^{c} - p_i \log_2 p_i \tag{2}$$

where c is the number of classification labels applied to the data set S and p_i represents the probability of label i within data set S.

Now that we have defined entropy, we can proceed to define information gain as it relates to entropy. Information gain is a statistical property that measures expected change in the entropy of S, with respect to a feature f. That is, when the entropy of S is reduced after partitioning on f, then partitioning around f has improved the task of classification. We will define information gain as a function [21]:

$$Gain(S, f) \equiv Entropy(S) - \sum_{v \in Values(f)} \frac{|S_v|}{|S|} Entropy(S_v) \tag{3}$$

where $Values(f)$ is the set of all possible feature values for f and S_v is the set of data points in S with the feature f equal to v. This is described as the entropy of S minus the sum of entropies of S

for each value of *f*. This represents the amount of information gained by partitioning the data set *S* around feature *f*. In terms of classical information theory, this represents the number of unnecessary bits when encoding an arbitrary data point in *S* provided the information about *f* [21]. Information gain is also known as Kullback-Leibler divergence [39].

As information gain approaches 1 when partitioning about a feature *f*, the more perfect division of the data about feature *f* becomes. For this reason, information gain makes the ideal statistical property when constructing decision trees [27]. The feature *f* at every given node in the tree with the highest information gain is the best feature to split the remaining data set.

Let us consider an example computation of information gain about a data set. For this example, Table 1 shows the number of instances in data set *S* with the feature *BodyTemp*, which can be described as $Values(BodyTemp) : \{HighFever, Normal\}$.

Table 1. BodyTemp training instances from *S*

	HighFever	**Normal**
Positive	6	3
Negative	2	3

From this table we can directly compute the entropies as follows:

$$Entropy(S) \equiv -\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14} \equiv 0.94029$$

$$Entropy(S, HighFever) \equiv -\frac{6}{8}\log_2\frac{6}{8} - \frac{2}{8}\log_2\frac{2}{8} \equiv 0.8118$$

$$Entropy(S, Normal) \equiv -\frac{3}{3}\log_2\frac{3}{3} - \frac{3}{3}\log_2\frac{3}{3} \equiv 1$$

From these entropies, we can also directly calculate the Information Gain about each of the features for *BodyTemp*:

$$Gain(S, BodyTemp) = Entropy(S) - \sum_{v \in \{HighFever, Normal\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$= Entropy(S) - \frac{8}{14} Entropy(S_{HighFever}) - \frac{6}{14} Entropy(S_{Normal})$$

$$= 0.94029 - \frac{8}{14} 0.8118 - \frac{6}{14} 1$$

$$= 0.4783$$

Information gain will be useful when interpreting the value of the trained lexicon in Chapter 6 and 7. Computing information gain over the set of words for each shaping factor will determine whether the trained lexicon is capable of classifying an arbitrary unlabeled report with the respective shaping factor.

CHAPTER 4

DATA SOURCES

Many of the techniques and algorithms used by this pipelined subjectivity classification system depend on specialized data sources. Every one of the data sources used for this project are manually annotated or categorize words from the English language. There is no requirement that this always be the case.

Some of the data sources are have overtly labeled polarities. That is, the data source directly indicates the most common polarity context: positive, negative, or neutral. The General Inquirer and MPQA lexicons specify a polarity classification label, which can immediately score the phrase. The prototype implementation uses these data sources as prior polarity labels, also referred to as "seeds," which are fed as input to other classifiers.

Several of data sources only provide associations between words, phrases, or groups of words and phrases. This is the primary function of the WordNet database and the implication of the data within a thesaurus [25]. For these data sources, a seeding data source is required for polarity classification. For example, to classify the root word from a single entry in a thesaurus, at least one of the synonyms must have a seeded label. Combined with a small seeding data source, these data sources often lead to the generation of a large subjectivity lexicon.

4.1 General Inquirer

The General Inquirer lexicon, or GI, is considered the gold standard for manually labeled subjectivity lexicons. The GI lexicon has been in existence for many years, dating to the early 1990's where it was only available to large IBM mainframes. Over time, the lexicon and toolset have evolved, from TrueBasic to Java support [27]. Modern desktop computers are capable of processing speeds unheard of when GI was originally developed, even evolving to an Internet centric usage model. Currently, the GI lexicon is available over the Internet, for free, as a raw text source and a web service [27, 11].

The GI lexicon is a mapping tool that maps a number of dictionary categories to a set of words. Currently, GI combines the Harvard IV-4 and Lasswell dictionary content-analysis categories and a few categories based on the work of (Semin and Fiedler) [15]. In total, there are approximately 182 categories with the negative category being the largest. Since we are only concerned with the subjectivity of words, the only categories used by this thesis are the positive and negative categories.

The GI lexicon data is available on the WebUse website [11] and is divided into individual categories often referred as tags. Each file contains a set of words associated to a specific tag. For example, the Pos tag data set, words used primarily in positive context, is stored within a single file, TAGPos.html [12]. There are other similar tags, such as Pstv, which may also be applied to the same word. Figure 3 demonstrates a sample of the Pos tagged data.

PLAUSIBILITY	Pos Noun Causal Eval |
PLAYFUL	IndAdj Pos Modif Nonadlt |
PLAYMATE	Pos Noun HU Role Nonadlt |
PLAYTHING	Pos Noun Object |
PLEASANT#1	Pos Modif EVAL Pleasur Pstv | adj: Agreeable, enjoyable
PLEASANT#2	Pos Modif Virtue Pstv | 0% adv: "Pleasantly"--agreeably

Figure 3. General Inquirer Pos sample

The start of each line is the capitalized form of the word followed by the tags and any available Parts of Speech (POS) labels. For example, the word "playmate" is a positive noun, also tagged as "nonadlt", "role", and "HU" (non-adult, role, and human) [14]. Some words also have multiple definitions, such as "pleasant" listed above, identified by the "#" and number following the word. Currently, there are 1914 Pos tagged words and 2293 Neg tagged words [12, 13].

4.2 Dictionaries

Another training data source that provides valuable information is a common dictionary. Often, words in the English language follow a pattern regarding subjectivity. This paper will demonstrate a proven method to exploit such patterns to build a seeding lexicon that is accurate when compared to the GI lexicon.

Since building a highly accurate and automatic lexicon discovery is one of the primary goals, dictionary-based techniques are highly desirable. They require no supervision and only a minimal set of rules to build a reliable lexicon. Furthermore, this technique is applicable to any dictionary training data.

A common approach, as implemented for this thesis, is to rely on affix patterns. That is, two words often share a similar affix pattern and form an Affix Seed Subjectivity Pairing (ASSP) when paired together. Each pair correlates to a positive and negative set of antonymous words.

To identify each set, an ASSP word pattern specifies one of the words in the pair as unmarked with the affix pattern, while the other word is overtly marked with an affix. The marked words are negative while the unmarked words are positive. An example ASSP would be honest / dishonest or imaginative / unimaginative. In those cases, the affix of "dis" and "un" mark the negative words in the ASSP. This thesis will target the 11 affix patterns listed within Appendix A [30].

This technique is effective when applied to dictionary data sources. The Affix Seed Lexicon, or ASL, has proven successful when applied to the Macquarie Thesaurus [1]. However, due to the inaccessibility of this data source, this thesis will use the distinct words from the Moby Thesaurus [15]. There is no restriction of application, as we may apply this algorithm to any arbitrary dictionary, though the accuracy is sensitive to the training data. Sections 5.3.2 and 6.3 will show that when applied against a much larger word list, such as a common Unix word list or the Moby Word Lists (SINGLE and COMPOUND) [15], the effectiveness of the technique is dramatically reduced.

The prototype implementation of this technique uses configurable affix patterns and training data sets; the only necessary component is a list of unlabeled words or phrases separated by a newline, accompanied by a set of affix seed patterns. Chapter 6 will also demonstrate that ASL will generate a small but accurate lexicon that is effective alone or useful as seed data to other subjectivity classifiers.

4.3 Thesauri

While dictionaries provide a valuable source of training data for automatic subjectivity classification, a thesaurus can provide similar benefits. It has been shown that thesaurus based techniques are effective, automatic, and require only common English reference data. In fact, a thesaurus is form of manually annotated semantic clustering of dictionary words. From a root word, a thesaurus clusters synonyms that all relate to a common concept, centered around a root word.

This manually annotated cluster makes a good source of training data for semantic orientation classification. If the root word is positive, most of the words associated to it should also be considered positive, as they are synonyms. Conversely, if the root word is negative, the associated word cluster is likely negative as well.

One such technique, referred to Macquarie Semantic Orientation Lexicon [1], or MSOL, has been shown to be effective against other subjectivity learning algorithms. Specifically, it was shown to be effective and extensive when compared to the SentiWordNet and Turney-Littman lexicons. The system presented here is capable of creating an even larger and more accurate subjectivity lexicon.

The Moby thesaurus is a CSV (comma-separated values) formatted data source, each word in a cluster comma delineated without quotations. Multi-word phrases are available as both root words and synonyms. Each line in the file represents a single cluster and the first phrase in the cluster is referred to as the “root word,” or the primary word with which the rest of the words relate.

The Moby thesaurus contains 30,259 root words, 103,305 unique words, a total word count (non-unique) of 645,505 words for an average cluster size of approximately 21 words, and is 24 megabytes in size.

4.4 MPQA

The Multi-Perspective Question Answering Subjectivity Lexicon, or MPQA, is the largest manually annotated data source. The MPQA lexicon is actually created from a variety of data sources, brought together to form a sizable collection of manual annotations. The multiple sources of data were compiled in the following order:

- List of subjectivity clues from (Riloff and Wiebe) [33]
- Collection of words from a dictionary and thesaurus
- Selection of words from the GI lexicon

Ultimately, this technique resembles many of the methods proposed by this thesis. The difference between the work done by (Wilson, Wiebe, and Hoffman) [19] to create the MPQA lexicon was done manually by human annotators, whereas this thesis proposes a semi-supervised learning techniques.

The utility of this data source comes from its high level of accuracy with the GI lexicon. In most cases, the polarity scores were directly copied from GI. This introduces bias, which is why an alternative evaluation lexicon is presented in Section 6.1. In addition, the MPQA lexicon is amongst the largest manually annotated data sources available, with about 8,000 total labeled words or phrases. Due to the high accuracy and large size, the MPQA data source is important and widely used for phrase level polarity classification and evaluation.

Each entry in the MPQA data set is comprised of the following fields:

- type – subjectivity clue
- length – number of words in the phrase
- word – word, stemmed word, or phrase
- POS – part of speech for the word
- stemmed – whether the word has been stemmed
- polarity – the non-contextual polarity clue

For this thesis, we are primarily concerned with the word and polarity fields. The current data set has only single worded phrases, thus length will always be 1. Each word may appear multiple times with a varying set of parameters. For example, the word "aberration" appears twice:

```
type=strongsubj len=1 word1=aberration pos1=adj stemmed1=n priorpolarity=negative
type=strongsubj len=1 word1=aberration pos1=noun stemmed1=n priorpolarity=negative
```

Figure 4. MPQA Sample Data

Notice that the first entry has a POS tag set to "adj" (adjective) and the second entry POS set to "noun." Each of those entries may be assigned a different value for the polarity clue; however, the case for "aberration" has the same polarity label assigned. When assigning multiple labels, the most common label is chosen. If there is no majority label, then the phrase shall be ignored as it is considered neutral.

4.5 WordNet

WordNet is a lexical database of semantic associations that is generally considered an authoritative tool for manual annotations regarding the English language. WordNet is capable of

expressing relationships between words, ontological information, even example usages and definitions. More importantly for this research, WordNet organizes the words in its database, of which there are currently 155,287 words, into sets of synonyms (or synsets), of which there are 117,659. In total, there are 206,941 word-sense pairings [16]. Due to the sheer size and maturity of the WordNet database, it makes a good training data source.

Querying the WordNet database returns a number of matches, instances, for the available sense keys. The sense key is the part of speech and the instance number indicates the definition number. For simplicity, we will refer to the word/sense key/instance number pairing in the following format: “word#sense#instance”. For example, the word and sense key pair “running” and “noun” will return 5 instances: “running#n#1” through “running#n#5”. Each instance may refer to a different meaning, context, or definition/example (also referred to as “glos”). Figure 5 shows a list of glosses for the five aforementioned instances of “running#n”. Figure 5 shows the full set of data retrieved from WordNet from the second instance, “running#n#2”.

running#n#1	(American football) a play in which a player attempts to carry the ball through or past the opposing team; "the defensive line braced to stop the run"; "the coach put great emphasis on running"
running#n#2	the act of running; traveling on foot at a fast pace; "he broke into a run"; "his daily run keeps him fit"
running#n#3	the state of being in operation; "the engine is running smoothly"
running#n#4	the act of administering or being in charge of something; "he has responsibility for the running of two companies at the same time"
running#n#5	the act of participating in an athletic competition involving running on a track

Figure 5. WordNet sample glosses

glos	the act of running; traveling on foot at a fast pace; "he broke into a run"; "his daily run keeps him fit"
syns	run#n#7, running#n#2
hype	locomotion#n#2
hypes	locomotion#n#2
hypo	dash#n#2
hypos	dash#n#2

Figure 6. WordNet sample data for running#n#2

Queries for each word contained in the WordNet database returns sense key pairs in a similar fashion. The notable thing about WordNet is that most words have a sense key applied, called “syns”, which is a set of words that belong in the same synset as the queried word. A synset is a set of words that relate to the same concept. As shown in Figure 6, other sense keys may be valuable, though this paper concerns itself with only the synset.

This thesis will present a method of exploiting these synsets to expand and classify subjectivity of words. Section 5.3.4 presents a method of searching WordNet to build a subjectivity lexicon.

CHAPTER 5

SEMI-SUPERVISED SUBJECTIVITY LEARNING ALGORITHMS

The first phase to processing jargon heavy corpora is to build a large and accurate subjectivity lexicon. This lexicon shall subsequently be used to determine whether there are any words that apply to the ASRS shaping factors to assist in causal analysis. The automatic learning methods presented here will discover phrases that apply to shaping factors that human annotators may have dismissed.

The proposed system of subjectivity classification is a system of PSC classifiers linked via their input and output lexicon data. Each PSC classifier will operate on their respective data source and alter the input lexicon using the input as a priori class label knowledge. The output of each PSC classifier can be fed directly to another subjectivity classifier. This creates a system of pipeline subjectivity classifiers that can be mixed in many combinations.

After every PSC classifier has performed a single training iteration, this prototype system can apply a boosting algorithm to further improve the accuracy against the baseline. A modification to the baseline AdaBoost algorithm enables the polarity score of each phrase to proportionally impact the training weight. Straight forward boosting assumes the training data set does not change, so as new training instances are discovered, the entire training set may need the training weights to be adjusted.

5.1 Overview

The implementation of the Semi-supervised Pipelined Subjectivity Classification system consists of reusable modules and a set of command-line tools tuned for flexibility. Each phase of pipeline processing is designed to take as input a lexicon of words and phrases, polarity scores, and accuracy weights. The output of each phase can be subsequently used as input to another pipeline phase. A single PSC algorithm represents an individual pipeline phase.

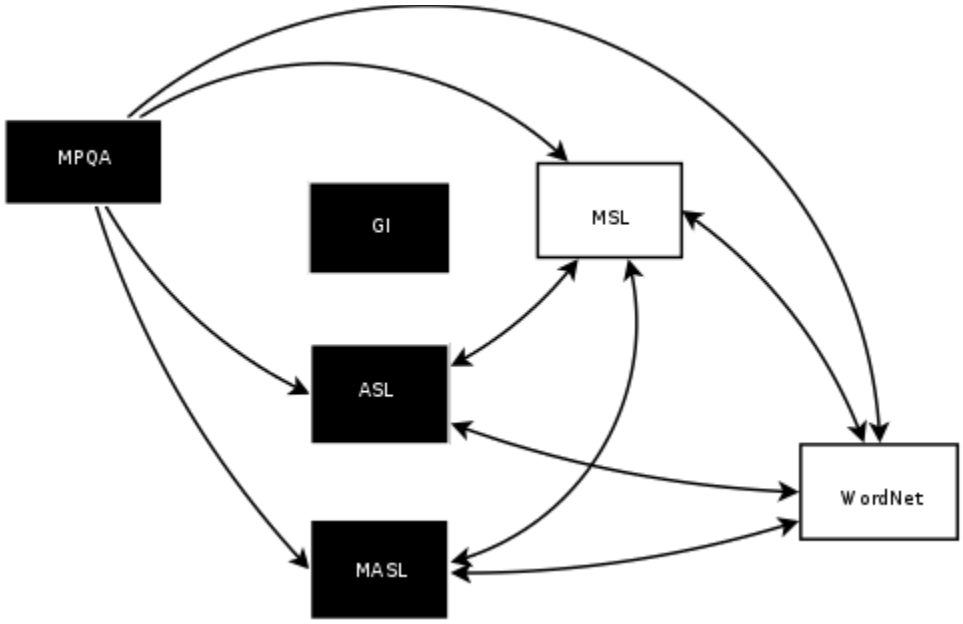

Figure 7. PSC Experimental Flow Chart

As shown in Figure 7, the pipeline classification system can be visualized as a directed graph. Each algorithm used in a pipeline execution represents a node. Solid colored nodes represent the starting (or seeding) nodes and the directed edges represent the paths an subjectivity lexicon may travel during classification. To determine the set of experiments performed for this project, start at each seeding node, the overt classifiers MPQA and ASL, and perform an acyclic traversal of the graph. The result is 21 possible pipeline execution paths, 1

isolated execution for the GI data source, and an additional 12 pipeline executions to increase the WordNet depth search from 2 to 3, for a total of 37 execution paths.

The lexicon files act as a communication mechanism between the various pipeline phases. Each file consists of a number of comma separated entries consisting of the following fields: word, polarity score, and accuracy weight. For example, "anger, -10, 0.00234323" would be a line from a lexicon file. This can be interpreted as the word or phrase "anger" with the polarity score of -10, has been assessed to have an accuracy weight of 0.00234323. A polarity score of a negative number implies the word is mostly seen in negative context, while a positive number implies the opposite.

The accuracy weight is used solely by the boosting algorithms and does not represent a real accuracy metric. That is, the accuracy of 0.00234323 does not mean that the word "anger" is accurate 0.2% of the time. As discussed in Sections 3.1 and 5.4.3, the initial weight of the word "anger" is adjusted as part of the boosting phase of the pipeline. Also, as discussed in Section 4.5.1, weights of currently known words are adjusted evenly as new words are discovered during a pipeline classification phase. The prototype implementation requires boosting to be applied after every phase in the pipeline, if boosting is applied at all. There is no requirement of the pipeline phases to utilize the accuracy weight, though only MSL and WordNet make use of the weights to determine polarity scores based on the word clusters.

5.2 Supervised Learning Algorithms

The only fully supervised learning algorithm considered for this project is the MPQA learning algorithm. Since GI was previously explained to be the testing data source, it will not be discussed further.

5.2.1 MPQA

As discussed in Section 4.4, the MPQA data is overly labeled as being either positive, negative, or neutral. The only information relevant to this algorithm is the *priorpolarity* field for each *word* in the data file. All of the words in the data source are single words (*word1*) and the strength of the subjectivity (*type*) is of no concern. Even if a word is *stemmed*, as long as it has a *priorpolarity*, the word is useful in seeding future PSC algorithms with valuable subjectivity information.

Each entry from the data source will be parsed and the word identified by the *word* field will have an assigned score based on the *priorpolarity* field. The score for the *word* is increased by 1 when the value for the *priorpolarity* field is set to "positive", and decreased by 1 when *priorpolarity* is set to "negative". This technique is a means of taking the consensus for words with different parts of speech and stemmed variations listed. This consensus technique is standard across every PSC presented in this paper.

In the current data source, there are 1255 words with multiple entries. There are also 571 words marked with a *priorpolarity* of "neutral"; those words will be ignored as only "positive" or "negative" have a polarity that is of value to this project.

5.3 Semi-supervised Learning Algorithms

Every algorithm, aside from the MPQA data source, labels the training data in a semi-supervised learning manner. That is, though words can be discovered in an automatic fashion, labels cannot be applied without prior information, such as labels or labeling mechanism. For

example, ASL needs the ASSP patterns to label dictionary words and MSL needs prior polarity scores to determine how to label each phrase thesaurus cluster.

Many machine learning problems have vast amounts of unlabeled data and minimal, if any, manually labeled training data sources. This is often the case because obtaining labeled data is infeasible and expensive. This is where semi-supervised learning techniques provide practical value. The algorithms discussed below need only a small set of prior labeled training instances to generate a large collection of labeled training data. In certain cases, the trained models from the labeled data sources can be more accurate than fully unsupervised learning techniques, though the resulting trained model may not be as comprehensive.

5.3.1 ASL – Affix Seed Lexicon

The dictionary-based method implemented for this thesis is the Affix Seed Lexicon, or ASL. This technique, based on the work of (Mohammad, Dorr, Dunne), will look through a dictionary and find ASSP words that match the 11 affix patterns discussed in Section 4.2 and listed in Appendix A.

This algorithm is considered a semi-supervised training method because, while the words can be marked either positive or negative without prior polarity scores, they require the marked and unmarked patterns to label the data. Without the ASSP patterns, the ASL algorithm cannot determine which words to mark as subjective and whether such words should have a positive or negative polarity score. The ASSP patterns are determined by a human annotator to be generate phrase pairings that frequently indicate positive and negative words.

The primary lexicon building tool, seed.pl, will accept a file that contains the affix pattern pairs as regular expressions. The first word listed is the positive unmarked word, which should

be placed entirely in a capture, and the second word is the marked negative word referencing the captured unmarked positive word. For example, to capture the “X, disX” ASSP, we will use the regular expression “^(.+)$,dis$1”. This expression reads in two parts:

- positive unmarked word: capture any character from start to finish
- negative marked word: “dis” followed by the unmarked positive word

Any number of these patterns can be used as long as they follow the rule that the unmarked positive word is referenced in the marked negative word and is an evaluative regular expression. Appendix A lists the implemented set of ASSP regular expressions to match the marked and unmarked words.

Since any word may be an unmarked positive word that pairs with a marked negative word, the runtime for this algorithm is $O(n^2m)$, where m is the number of affix patterns and n is the number of words in the dictionary data source. That is, for every affix pattern, every word must be compared to every other word.

The data source chosen to train this PSC algorithm is the set of unique words taken from the Moby Thesaurus. However, any word list file may be used as a data source, as demonstrated by the Moby (wordlist) Affix Seed Lexicon. The Moby Thesaurus has 103,305 unique words and phrases.

5.3.2 MASL – Moby (wordlist) Affix Seed Lexicon

A slight variation on the ASL algorithm is the Moby (wordlist) Affix Seed Lexicon. The purpose of this variation is to demonstrate the sensitivity of the input data source. This experiment uses the single word and compound phrases as identified in the Moby Word List

files. These word lists are much larger in size and initially appeared to be a more valuable data source. The single word list file contains 354,983 unique words and the compound phrases file contains 256,772 unique phrases, resulting in a data source of 611,755 unique words and phrases. This data source is nearly 6 times larger than the unique words list taken from the Moby Thesaurus.

The only difference is the data source passed to the unmodified ASL algorithm. As demonstrated in Section 6.3, we will see that the result of using the Moby word list files lowers the accuracy compared to the GI lexicon. Due to the low performing nature of this algorithm, we will not use this PSC in further experiments.

5.3.3 MSL – Moby Seed Lexicon

Based on the Macquarie Semantic Orientation Lexicon, the Moby Seed Lexicon, MSL, algorithm follows the work of Mohammad et al. [1]. Mohammad et al. proposed a system of classification based on the Macquarie Thesaurus. Since the Macquarie Thesaurus is a commercial (non-free) data source, the Moby Project Thesaurus functions as an alternative for this thesis.

The algorithm works by iterating over every word cluster in the Moby thesaurus, delimited by a single line. If a majority of the phrases in the cluster have a positive label, polarity score greater than 0, then the polarity score of every phrase in the cluster will be increased by 1 (which indicates an additional positive label). Conversely, the polarity score is decreased by 1 if the majority label is negative. Each word or phrase may appear multiple times across the thesaurus clusters; therefore, the most common label at the end is taken from the positivity or negativity of the final polarity score.

Since words and phrases may appear as both root words and as synonyms for other root words, order of processing should not affect scores of words processed near the end. This is done because newly discovered words early in the process should not incorrectly alter the classification of words later in the process. For example, the word “modern” appears as a synonym to the root phrase “a la mode” as the second line in the file, yet “modern” appears as a synonym for 62 words. For this reason, we must store the polarity scores, and weights, in a separate structure and use the prior subjectivity lexicon as the basis for the current labels.

5.3.4 WordNet

The WordNet subjectivity classifier is also a thesaurus based learning method. By traversing through the synsets in WordNet, this algorithm can discover new words and label them based on prior labels. This classifier is similar to the graph traversal method presented by [30].

The algorithm works by iterating over all of the words and phrases in the prior subjectivity lexicon, and for each word, this algorithm will traverse WordNet in a depth first search (DFS) to find other related words. The set of synonyms taken as the related word cluster come from the “syns” sense key. The resulting set of related words, after searching to a depth of *d*, is conceptually the same as the manually clustered phrases from a thesaurus. The root word in this case is the word that starts the depth first search.

To start by searching for a word, for example “pain”, each supported part of speech (POS) must be queried. Since this algorithm is broadly searching without POS labels, it will check noun, verb, adjective, and adverb. That is, the terms “pain#n”, “pain#v”, “pain#a”, and “pain#r” will be searched for related words. When querying terms without specifying an instance

number, as described in Section 4.5, the initial results are typically instances of the word and POS. For example, “pain#n” will find “pain#n#1”, “pain#n#2”, and so forth.

Each of the synset words returned will trigger a recursive call with a reduced remaining depth, emulating a DFS search. A discovered synset word that lacks an instance number does not decrease the search depth because only the synset instances that pertain to a specific glos are of value. Instances can be seen as leaves in the DFS search through WordNet.

Once a full set of related words has been discovered, the root word will either increase or decrease depending on majority of the polarity scores in the set of related words. If the related words have a mostly positive polarity score, which is a score that sums to greater than 0, the entire cluster of related words will have their polarity score *increased* by 1. When the cluster of words has a mostly negative polarity score, which is a score that sums to less than 0, the entire cluster of related words will have their polarity score *decreased* by 1.

To ensure that the order of processing the phrases from the prior lexicon does not label the current polarity lexicon incorrectly, the labels are applied to a new structure without altering the prior lexicon structure. This is the same method and reasoning as done for the MSL classifier.

When querying WordNet for word cluster discovery, the sense keys are a critical step. Table 2 below shows the sense keys used by this project for the various parts of speech; we are primarily concerned with similar words only.

Part of Speech	*Sense Key*	*Meaning*
nouns	syns	synset words
verbs	syns	synset words
adjectives	syns	synset words
adverbs	also	also see

Table 2. WordNet query sense keys

5.4 Boosting

Many of the PSC pipeline combinations have a low performance, as shown in Section 6.3. For this reason, boosting provides a proven means of improving performance of these weak classifiers. In fact, that is the exact purpose of boosting algorithms, such as AdaBoost. A flexible boosting scheme is presented to further improve the accuracy of the existing pipeline classification system without sacrificing lexicon size.

As discussed in Chapter 3, the AdaBoost algorithm adds emphasis on the misclassified examples while deemphasizing the correct classifications. The AdaBoost algorithm is performed after each iteration of a pipeline classifier. That is, once a phase in the pipeline execution phase has completed, it boosted by the AdaBoost module and then the same pipeline classifier is restarted. This is continued for a predetermined number of iterations, which correlates to *M* in Figure 1.

The reason the boosting is applied to the current phase in succession is to increase the accuracy as much as feasible before passing its resulting lexicon to another classifier as the a priori lexicon. This method is employed over boosting a complete pipeline execution to increase the performance per pipeline phase and reduce the number of required iterations. While the same subjectivity lexicon should eventually be reached, the prototype system presented here will only demonstrate inter-phase boosting.

There are many matters to consider when boosting an individual set of PSC classifiers. Internal data must be reset between iterations as to not introduce bias. For algorithms that discover new words from their respective data source, weights need adjustment upon discovery. Lastly, the number of iterations, M, necessary should be an appropriate trade-off between time of execution and accuracy improvement. This section will explain these matters further.

5.4.1 Reweighting

Since boosting is performed for each pipeline phase, new words may be discovered between each iteration of a phase. That is, when an PSC classifies a subjectivity lexicon, new words may be discovered from its respective data source, their weight must be initialized to accommodate the weight of the existing lexicon. In fact, the entire trained subjectivity lexicon must adjust to accommodate the newly discovered words.

As shown in step (1) of Figure 1, we want to initialize the weights proportional to the size of the new additions. The new words will have equal initial weights proportional to the size of the new lexicon and prior words will be have their weights recomputed to match their proportion to the new lexicon. Equation 4 demonstrates the formula for recalculating the weights.

$$w(x_i) = \begin{cases} w_{m-1} * \dfrac{|x_\rho|}{|x_\varphi|}, & x_i \; is \; preexisting \\ \dfrac{|x_\Delta|}{|x_\Delta| * |x_\varphi|}, & x_i \; is \; new \end{cases} \tag{4}$$

where x_i is the lexicon instance, w_{m-1} is the prior weight for instance x_i, x_ρ is the prior lexicon, x_φ is the lexicon after PSC classification, and x_Δ is the newly discovered lexicon phrases. When the size of the lexicon does not change, reweighting is unnecessary.

For example, if there are 90 prior words and 10 new words, then the weights of the prior words will be weighted to 90% of their prior weight and the new words will all be initialized to have 0.01 weight.

5.4.2 InvBoost – AdaBoost Alternative

The weight of each word can be considered a measure of accuracy after each iteration of classification and boosting. As AdaBoost was shown in Section 3.1, inaccurately classified subjective words have their weight increased and accurately classified words have their weight decreased. This is a desirable feature that can be used to reduce the effects of frequently misclassified words. Since many of the pipeline classifiers are cluster-based methods, words that are misclassified can in effect trigger more misclassifications. For example, if the word "damage" is misclassified as being a positive word, then it will increase the polarity score of "harm" and "impairment", triggering a positive classification to words that are clearly negative.

Directly applying the weight assigned by the AdaBoost algorithm to the polarity score has the opposite desired effect: misclassified instances will have higher polarity scores because AdaBoost increases the weight. Either the polarity score must adjust inversely proportional to the weight or the weight adjusts inversely proportional to AdaBoost. Either the weight must apply inversely to the polarity score or the weight will have to adjust proportionally inverse. That is, Equation 5 shows the two weight update rules, where ω_i represents the weighted polarity score.

$$\omega_i = \frac{x_i}{w_i}$$
$$\omega_i = x_i * w_i \tag{5}$$

Alternatively, the weight adjustment during boosting can be assigned in the opposite manner. Equation 6 shows the alternative update that decreases the weight of misclassifications and increases the weight for correct classifications. Notice how this weight update rule is the inverse of the weight update rule for AdaBoost shown in Figure 1. With this weight update rule, we can produce weight adjusted polarity score that can be directly summed over a synset, which also emphasizes correct classifications and de-emphasizes misclassifications.

$$w_i = w_i * e^{\alpha_m * I(y_i \neq h_i(x_i))}, i = 1, 2, \ldots N$$
$$I(y_i \neq h_i(x_i)) = \begin{cases} 1, & if\ y_i\ is\ correctly\ classified \\ -1, & if\ y_i\ is\ misclassified \end{cases} \tag{6}$$

This alternative method, InvBoost (Inverse AdaBoost), will be shown to be an effective means to improving the performance and increase the size of the resulting lexicon.

5.4.3 Iterations

Since algorithms such as AdaBoost are designed to improve a weak classifier by training a number of weighted hypotheses, as shown in Section 3.1. Determining the number of iterations necessary is not a trivial task since training weak classifiers may be the most time consuming task of classification.

One of two means typically determines the appropriate number of iterations, *M* as shown in Figure 1:

- Automatically stop when either the accuracy or performance exceeds an acceptable threshold or the change between iterations is below a threshold.
- Experimentally, balancing training time and accuracy

For this project, an appropriate *M* was determined experimentally. An arbitrary number of iterations that demonstrates an improvement over the baseline method was chosen. Three iterations of boosting showed marked improvements and shall be sufficient for demonstration purposes, though there is no limit in the prototype implementation.

It is equally important to know when to perform boosting. For a classification system such as this, there were two options:

- Boost after a complete execution of the pipeline classification, iterating *M* times.
- Boost after every phase of execution in the pipeline classification, *M* times, before proceeding to the next phase of pipeline classification.

The prototype system demonstrates the second method because of the dependent nature of the hypotheses generated by each of the PSC classifiers. Since the output of one phase is input directly to the next phase, each phase should be as accurate as possible before continuation.

5.5 Complications

There are a large number of words in the English language that can be deemed neutral in every possible context. Such words should have no bearing on the classification of other words since the pipeline classification system does not account for neutrality. Chosen words come from a several parts of speech:

- prepositions – such as of, to, in, for, with

- conjunctions – such as and, or, but, yet, neither
- pronouns – such as he, she, it, himself

These words were added to a file referred to as the "stop words" file. During training, if a word is discovered that is an exact match to a stop word, then it will be ignored and cannot influence the discovery or classification of other words. Notice that this check is for exact matches only. Stop words may be multi-word phrases just as the rest of the words discovered during training. However, a stop word may be used within another phrase. For example, a phrase "plunged in grief" has stop word "in", though the entire phrase has a strongly negative connotation.

Stop word lists are commonly used for search engines, databases, and other Natural Language Processing tasks that must omit filler words. The list of words used by the prototype system is a modified SMART stop word list. Appendix B shows all of the words removed from [25], as a human annotator deemed them to *possibly* be subjective. The list originally included words found in the GI lexicon, so these were removed as well.

The result of removing stop words upon discovery leads to lightly smaller trained lexicon. In some cases, the accuracy decreased slightly, though most trained lexicons improved accuracy when compared against the GI lexicon.

5.6 Evaluation Method

In order to understand the quality of a trained lexicon, a standard measure must be employed that provides insight to the improvements over the baseline. The typical measure of performance is the F-1 score, which is a harmonic mean of the precision (ability to reproduce the

results under the same conditions) and the recall (portion of relevant all data points in S returned by a query) [7]. For the purposes of this thesis will only report the accuracy and F-1 scores, though other metrics, such as discriminate power and youden, can be calculated.

All of the trained lexicons will be evaluated against the PCMA lexicon, discussed in Section 6.1. The trained lexicons, which do not have boosting applied, will be evaluated against the baseline GI lexicon. The trained lexicons with boosting applied the GI lexicon as the validation data set, to avoid bias, and can only be tested against the PCMA lexicon.

CHAPTER 6

SUBJECTIVITY CLASSIFICATION RESULTS

To find the largest and most accurate subjectivity lexicon, all possible experiments from the PSC flow chart in Chapter 5 formed the set of experiments. In total, 37 experiments were performed, once non-boosted, and once boosted with both AdaBoost and InvBoost. There are 222 experiments performed, each providing a lexicon that can be evaluated against the GI lexicon and the PCMA lexicon. This chapter will cover the need for an alternative manually labeled lexicon, the performance of non-boosted algorithms, and the performance of the boosted algorithms. Lastly, this chapter will discuss the lexicon selected for analysis of the ASRS corpus.

6.1 PCMA – Pipeline Classification Manual Annotations

Since the MPQA data source was partially built based on the GI lexicon, we must use an alternative lexicon during testing in conjunction with the GI lexicon. That is, we will evaluate the accuracy of each pipeline execution against the GI lexicon and this manually created lexicon. This lexicon will be referred to as the Pipeline Classification Manual Annotations lexicon, or PCMA. It was created by sampling 500 words and phrases from a lexicon created during development. Each of the words or phrases was assigned a polarity score by a human annotator. The human annotator had access to the definition of the word in the event that the word is new to their personal vocabulary level.

After the first human annotator labeled all 500 words, an additional human annotated the lexicon a second time, ensuring there is a high level of agreement concerning the labels. (Artstein and Poesio) developed this technique [35]. The PCMA lexicon was determined to have 92% agreement between both annotators, ensuring that the labels are reliably accurate.

Lastly, the PCMA lexicon will serve as the testing data when performing the boosting techniques discussed in Section 5.3. Since the GI lexicon is much larger than the PCMA lexicon, and many of the pipeline executions involve the MPQA technique, the GI lexicon will serve as the training data during boosting. This also ensures that bias is not introduced as part of the training phase since the MPQA data set originates from the GI lexicon.

6.2 Configuration

To automatically build each of the subjectivity lexicons needed for experimentation, a set of configuration files was developed that set common options to the various PSC algorithms. Figure 8 demonstrates one of the configuration files.

```
{
    "algo": [ "MPQA", "WordNet", "ASL", "MSL" ],
    "mpqa": "data/MPQA/subjclueslen1-HLTEMNLP05.tff",
   "affix": "data/affixes.txt",
    "dict": [ "data/moby/dict/moby.dat" ],
    "thes": "data/moby/thes/mthesaur.txt",
   "depth": "3"
}
```

Figure 8. Example JSON Configuration

Notice the configuration file is a JSON formatted file. It shows the following options set:

- Order of PSC: MPQA, WordNet, ASL, MSL
- MPQA data source for the MPQA PSC

- ASSP pattern file for the ASL PSC
- Dictionary file, unique words from the Moby Thesaurus, for the ASL PSC
- Thesaurus file, Moby Thesaurus, for the MSL PSC
- WordNet DFS search depth

6.3 Pre-Boosting Subjectivity Classification

The first set of experiments involves building the lexicons for each possible pipeline execution at shown in the graph in Chapter 5. Each experiment represents a single path through the pipeline execution graph; each node visited represents a classifier applied to the prior subjectivity lexicon in the order of visitation. Table 3 lists each experiment and its respective performance. Note that "WNdX" refers to WordNet with a DFS search depth of X.

Experiments are evaluated against two reference lexicons: GI and PCMA. For each of the reference lexicons, the performance of an individual experiment is determined based upon: the number of words in common, accuracy, and F-1 score.

Pipeline	Size	vs. GI		
		Common	Accuracy	F-1 Score
ASL	8189	1025	78.4390	83.6417
MASL	56696	2168	62.0387	72.8652
ASL + MSL	87528	3276	63.2173	70.3713
ASL + MSL + WNd2	107598	3239	62.8898	71.2165
ASL + MSL + WNd3	107581	3223	63.1089	71.3976
ASL + WNd2	15118	1847	69.8971	76.2799
ASL + WNd2 + MSL	98792	3376	61.9076	69.8122
ASL + WNd3	15151	1848	70.1299	76.3496
ASL + WNd3 + MSL	98712	3365	62.1100	70.0071
MPQA	6436	2863	98.4282	98.1053
MPQA + ASL	12720	2875	96.8696	96.4926
MPQA + ASL + MSL	94337	3198	91.4009	90.9331
MPQA + ASL + MSL + WNd2	113366	3359	91.3367	90.8057
MPQA + ASL + MSL + WNd3	113370	3360	91.4881	90.9608
MPQA + ASL + WNd2	22892	3196	94.4931	93.9643
MPQA + ASL + WNd2 + MSL	101113	3469	92.9374	92.3366
MPQA + ASL + WNd3	22910	3193	94.4879	93.9436
MPQA + ASL + WNd3 + MSL	101115	3468	92.9066	92.3029
MPQA + MSL	90391	3215	92.3484	90.8618
MPQA + MSL + ASL	88765	3237	92.6784	91.5266
MPQA + MSL + ASL + WNd2	107490	3375	92.2074	91.0024
MPQA + MSL + ASL + WNd3	107491	3374	92.2644	91.0647
MPQA + MSL + WNd2	108711	3351	91.7935	90.2378
MPQA + MSL + WNd2 + ASL	109159	3387	91.7036	90.2802
MPQA + MSL + WNd3	108694	3354	91.8306	90.2837
MPQA + MSL + WNd3 + ASL	109146	3389	91.7085	90.2869
MPQA + WNd2	14379	3145	96.4070	95.7310
MPQA + WNd2 + ASL	19813	3211	96.0137	95.4023
MPQA + WNd2 + ASL + MSL	98489	3478	93.8183	93.0801
MPQA + WNd2 + MSL	98805	3485	93.3716	92.2974
MPQA + WNd2 + MSL + ASL	97185	3462	93.9341	93.0093
MPQA + WNd3	14452	3148	96.3469	95.6620
MPQA + WNd3 + ASL	98505	3481	93.8523	93.1234
MPQA + WNd3 + ASL + MSL	98828	3482	93.4520	92.3797
MPQA + WNd3 + MSL	97201	3465	93.9394	93.0140

Table 3. Pre-Boosting PSC Results vs. GI

Pipeline	Size	vs. PCMA		
		Common	Accuracy	F-1 Score
ASL	8189	86	89.5349	88.0000
MASL	56696	158	77.2152	79.5455
ASL + MSL	87528	411	60.5839	65.3846
ASL + MSL + WNd2	107598	446	59.6413	64.5669
ASL + MSL + WNd3	107581	446	59.6413	64.5669
ASL + WNd2	15118	119	84.8739	83.9286
ASL + WNd2 + MSL	98792	440	59.7727	64.9505
ASL + WNd3	15151	120	85.8333	84.6847
ASL + WNd3 + MSL	98712	440	60.4545	65.4762
MPQA	6436	58	98.2759	96.7742
MPQA + ASL	12720	128	92.1875	89.5833
MPQA + ASL + MSL	94337	439	80.4100	77.2487
MPQA + ASL + MSL + WNd2	113366	464	82.5431	79.1774
MPQA + ASL + MSL + WNd3	113370	462	82.9004	79.5866
MPQA + ASL + WNd2	22892	186	91.9355	89.3617
MPQA + ASL + WNd2 + MSL	101113	449	82.8508	79.4667
MPQA + ASL + WNd3	22910	186	91.9355	89.3617
MPQA + ASL + WNd3 + MSL	101115	449	82.6281	79.2553
MPQA + MSL	90391	436	82.1101	75.6250
MPQA + MSL + ASL	88765	426	82.8638	76.5273
MPQA + MSL + ASL + WNd2	107490	458	84.9345	78.8991
MPQA + MSL + ASL + WNd3	107491	456	85.3070	79.3846
MPQA + MSL + WNd2	108711	464	84.4828	78.0488
MPQA + MSL + WNd2 + ALS	109159	468	84.4017	78.2090
MPQA + MSL + WNd3	108694	464	84.4828	78.0488
MPQA + MSL + WNd3 + ASL	109146	467	84.3683	78.2090
MPQA + WNd2	14379	114	93.8596	90.1408
MPQA + WNd2 + ASL	19813	174	92.5287	89.7638
MPQA + WNd2 + ASL + MSL	98489	446	84.3049	80.2260
MPQA + WNd2 + MSL	98805	450	84.6667	78.7692
MPQA + WNd2 + MSL + ASL	97185	442	85.2941	79.6238
MPQA + WNd3	14452	116	93.9655	90.4110
MPQA + WNd3 + ASL	98505	443	84.4244	80.3419
MPQA + WNd3 + ASL + MSL	98828	449	85.0780	79.3846
MPQA + WNd3 + MSL	97201	440	85.9091	80.5031

Table 4. Pre-Boosting PSC Results vs. PCMA

The ASL method was surprising effective against the GI lexicon. While the ASL algorithm was effective alone, it was also effective in combination with other PSC algorithms. When used early in the pipeline, the ASL PSC found approximately 6,000 words with decent performance. When used later in the pipeline, ASL discovers fewer words, lowers the performance, and in some cases, reduces the total lexicon size. For example, the MPQA + ASL + WNd2 experiment discovers approx. 7,500 additional total words, approx. 50 GI words, and reduces performance against GI by approx. 2%. The research from [1] proved valuable as most of the words classified by this method are accurate.

The MASL method, ASL used with the Moby word list files, discovers significantly more words than ASL at the cost of a large performance decrease (16% lower against GI). Due to the low performance and classification method, (boosting will not improve the MASL classifier), further experiments did not involve this PSC.

The MSL method discovers more words than any other classifier, regardless of the stage of execution in the experiment. However, unless it is combined with a high quality classifier, its performance is less than desirable. For example, ASL + WNd2 + MSL experiment, when compared to the ASL + WNd2 experiment, shows approx. 83,000 words discovered though the performance 84.8% accuracy to 59.7% (versus GI). However, the MPQA + ASL + WNd2 + MSL experiment, compared to MPQA + ASL + WNd2 experiment, discovers approx. 80,000 and only reduces the performance by approx. 9%. The MSL algorithm from [1] has also proven valuable in certain circumstances.

The WordNet method has been shown to be successful. The WNd2 method typically discovers approx. 15,000 words and usually only reduces the performance by 1-5%. Though it

may discover fewer words than the other methods, it's able to ability to maintain the performance before modifying the prior lexicon. That being said, the WNd3 (WordNet depth = 3) method provides little benefit. The discovered word count is insignificant and the performance did not change in most experiments.

The experiments involving MPQA are by far the best performing pipeline combinations. This PSC performs well against both the GI and PCMA reference lexicons. For example, the MPQA + ASL + MSL + WNd2 experiment produces a lexicon of comparable size to ASL + MSL + WNd2, though the performance goes from 59.6% to 82.4% when evaluated against PCMA. In fact, every experiment that involves MPQA has improved performance over the experiments excluding MPQA. This PSC has proven to be valuable source of seeding data.

6.4 Boosted Subjectivity Classification

After collection of the initial results, each pipeline execution combination was boosted to improve performance. The same options and input data sources were applied to perform training and to collect the boosted results. As discussed in Section 5.4, each experiment was trained with the GI lexicon and evaluated against the PCMA lexicon. Boosting was carried out to the third iteration, determined experimentally to be sufficient.

Pipeline	Size	vs. PCMA		
		Common	Accuracy	F-1 Score
ASL	8189	86	89.5349	88.0000
ASL + MSL	99868	432	50.9259	60.8856
ASL + MSL + WNd2	123750	471	49.0446	59.5960
ASL + MSL + WNd3	123742	471	49.0446	59.5960
ASL + WNd2	26829	174	72.4138	72.4138
ASL + WNd2 + MSL	107537	458	54.3668	62.0690
ASL + WNd3	26841	174	72.4138	72.4138
ASL + WNd3 + MSL	107592	458	54.5852	62.3188
MPQA	6436	58	98.2759	96.7742
MPQA + ASL	12720	128	92.1875	89.5833
MPQA + ASL + MSL	100240	441	77.5510	72.7273
MPQA + ASL + MSL + WNd2	122377	475	78.1053	73.4694
MPQA + ASL + MSL + WNd3	122364	473	78.4355	73.8462
MPQA + ASL + WNd2	31648	209	87.0813	83.8323
MPQA + ASL + WNd2 + MSL	107811	469	78.8913	74.5501
MPQA + ASL + WNd3	31637	209	87.0813	83.8323
MPQA + ASL + WNd3 + MSL	107750	469	78.4648	74.1688
MPQA + MSL	101344	456	75.2193	65.8610
MPQA + MSL + ASL	99585	435	78.6207	70.2875
MPQA + MSL + ASL + WNd2	122181	472	80.7203	73.4694
MPQA + MSL + ASL + WNd3	122165	470	81.0638	73.9003
MPQA + MSL + WNd2	123335	481	77.5468	68.4211
MPQA + MSL + WNd2 + ALS	123566	487	77.8234	69.3182
MPQA + MSL + WNd3	123308	481	77.5468	68.4211
MPQA + MSL + WNd3 + ASL	123559	486	77.7778	69.3182
MPQA + WNd2	25086	148	85.8108	80.3738
MPQA + WNd2 + ASL	29978	203	87.1921	83.1169
MPQA + WNd2 + ASL + MSL	105966	452	79.4248	73.1988
MPQA + WNd2 + MSL	107440	466	75.9657	67.4419
MPQA + WNd2 + MSL + ASL	105954	451	78.7140	71.2575
MPQA + WNd3	25109	148	86.4865	81.1321
MPQA + WNd3 + ASL	105934	451	79.8226	73.7752
MPQA + WNd3 + ASL + MSL	107409	465	75.9140	67.2515
MPQA + WNd3 + MSL	105901	450	78.8889	71.2991

Table 5. AdaBoost (M=3) PSC Results

Pipeline	Size	vs. PCMA		
		Common	Accuracy	F-1 Score
ASL	8189	86	89.5349	88.0000
ASL + MSL	100154	435	66.2069	68.5225
ASL + MSL + WNd2	122641	463	67.6026	70.1195
ASL + MSL + WNd3	122611	463	67.3866	69.9801
ASL + WNd2	26798	172	76.7442	75.9036
ASL + WNd2 + MSL	107211	458	66.3755	68.4426
ASL + WNd3	26806	172	76.1628	75.4491
ASL + WNd3 + MSL	107231	459	66.2309	68.3027
MPQA	6436	58	98.2759	96.7742
MPQA + ASL	12720	128	92.1875	89.5833
MPQA + ASL + MSL	100206	446	80.7175	77.4869
MPQA + ASL + MSL + WNd2	122090	475	82.5263	78.9873
MPQA + ASL + MSL + WNd3	122079	474	82.7004	79.1878
MPQA + ASL + WNd2	31642	208	87.5000	83.9506
MPQA + ASL + WNd2 + MSL	107765	465	83.8710	80.3150
MPQA + ASL + WNd3	31635	209	88.0383	84.8485
MPQA + ASL + WNd3 + MSL	107747	466	83.9056	80.4178
MPQA + MSL	101556	456	80.4825	74.9296
MPQA + MSL + ASL	99537	445	81.3483	75.8017
MPQA + MSL + ASL + WNd2	121869	475	82.9474	77.4373
MPQA + MSL + ASL + WNd3	121862	474	83.1224	77.6536
MPQA + MSL + WNd2	123098	482	82.3651	76.5840
MPQA + MSL + WNd2 + ASL	123358	485	82.4742	76.9648
MPQA + MSL + WNd3	123081	482	82.3651	76.5840
MPQA + MSL + WNd3 + ASL	123355	485	82.2680	76.7568
MPQA + WNd2	25113	149	85.2349	79.2453
MPQA + WNd2 + ASL	30003	203	87.1921	82.8947
MPQA + WNd2 + ASL + MSL	105963	455	83.5165	78.9916
MPQA + WNd2 + MSL	107480	465	81.5054	75.2874
MPQA + WNd2 + MSL + ASL	105896	458	82.5328	76.7442
MPQA + WNd3	25150	148	86.4865	80.7692
MPQA + WNd3 + ASL	105956	453	84.1060	79.7753
MPQA + WNd3 + ASL + MSL	107453	463	81.8575	75.7225
MPQA + WNd3 + MSL	105856	456	82.8947	77.1930

Table 6. InvBoost (M=3) PSC Results

For the M=3 boosting results, the AdaBoost algorithm reduced the performance in every experiment involving a PSC that considered the boosting weights during classification. The InvBoost experiments also demonstrate a reduced performance, though less than AdaBoost. For example, AdaBoost reduces the MPQA + ASL + MSL + WNd2 experiment results from 82.5431% to 78.1053%, whereas the performance of InvBoost is 82.5263%, which is comparable to the non-boosted version. This reduction in performance, even by InvBoost indicates the inter-phase subjectivity lexicon is noisy.

However, in nearly every experiment, the lexicon expanded in size. This is due to the repeated classification of a PSC while boosting. When each PSC, such as the MSL or WordNet, restarted after boosting, they had a larger prior lexicon from the previous iterations. As the lexicon expanded between iterations, so did the number of word clusters and synsets containing labeled training data. The rate of growth depends on the size of the word clusters and synsets. This was a logical yet unexpected result.

The lexicon chosen as the ASRS Reference Lexicon, ARL, was a result from the MPQA + MSL + WNd2 + ASL with InvBoost(M=3) experiment. This lexicon was chosen as it was one of the largest and highest performing lexicons available. While the rest of the experiments do not depend on which lexicon was chosen as the ARL, the results are tightly coupled with the ARL.

6.5 Lessons Learned

The original paper presented by [1] demonstrates that the MSOL and ASL methods can be highly effective, though as shown above, they are sensitive to the input data sources. The WordNet method is also effective, though it trains a smaller lexicon. There were many complications building a subjectivity lexicon that could be trusted to be accurate.

The time of execution for each of the experiments presents a minor hurdle to classification. It took over 24 hours to collect all of the results listed in Tables 3 through 6. While this is not prohibitively long, it does force implementers to consider external factors, such as implementation language, hardware, and size of input data sources. However, if only training a specific experiment, such as the ARL, it took approximately 10 minutes on current hardware. Discovering stop words late in experimentation meant that experiments had to be performed several times over to ensure the proper sets of stop words were excluded.

Implementation of AdaBoost and InvBoost was difficult to verify. Much time was wasted determining and verifying the best way to apply the weights. Most significantly, a minor error in calculation may be imperceptible, as the early system neglected to normalize the weights to a probability distribution shown in step (2b) of Figure 1.

CHAPTER 7

JARGON HEAVY CORPORA

Domain specific languages present a different set of problems that would not otherwise be the present with more traditional corpora, such as the New York Times Annotated Corpus [20]. The ASRS corpus uses a jargon heavy language that makes traditional natural language processing difficult. The Natural Language Processing community has not explored usage of jargon heavy corpus extensively. This chapter will demonstrate that even though the corpus has a large number of words undefined by most English reference sources, we are able to utilize such a corpus for learning models.

Before any text containing large amounts of jargon words and phrases can serve as a training data source or corpus, we must first identify key features for a classifier. Posse et al. defined a set of shaping factors [34], a set of influential factors based on contextual information, that identify the source of one or more problems for each report within the ASRS corpus [9]. This thesis aims to show that the reports can be classified a shaping factors based on the subjectivity lexicon trained in Chapter 5 and 6.

7.1 Issues with the ASRS Corpus

The initial problem with the ASRS reports is the formatting of the report itself. Each report is represents a dictated event describing a problem that occurred before, during, or after a flight. Since the reports are submitted electronically or retyped from mailed submissions,

punctuation is rarely correct, words are often misspelled, sentence fragments are common, and misuse of capitalization is frequent. In comparison, the New York Times Annotated Corpus, created from New York Times publications, is properly formed, well written, and a highly regarded publication.

Another key obstacle to causal analysis of the ASRS corpus is the frequent usage of jargon words, or words that not typically found in dictionaries or thesauri. For example, the phrases "MECHS" and "nosewheel" are technical and specialized words not found by any of the reference dictionaries. Figure 9 demonstrates usage of jargon terms taken directly from the ASRS corpus.

Immediately descended below the lower layer to circling minimums instead of waiting to the outer marker.
Received the Automatic Terminal Information Service at IAD with the following Notice To Airmen; taxiway whiskey closed between whiskey one and whiskey ten.

Figure 9. Example ASRS sentances

Since most of the vernacular used in the reports relates to actions taken by a pilot while piloting a plane, there is a large number of words that have special meaning in context. Many of these terms refer to jargon not frequently seen outside of the context of aviation. From Figure 9, we can see the terms "circling minimums", "outer marker", "IAD", "whiskey one" and "whiskey ten" all have special meaning to aviators (pilots and air traffic controllers).

To accompany the ASRS corpus, the ASRS training data set will serve a valuable purpose during this phase. The ASRS training data consists of reports that were manually labeled

with one or more shaping factors. This information was utilized in conjunction with the Basilisk framework by (Mohammad, Ng, Khan). Each entry in the training data consists of the following comma delimited information:

- ASRS Report ID
- The full unprocessed report
- Semi-colon delimited Shaping Factors

The shaping factors listed at the end of the training report were determined to best describe the source of the problem discussed in the unprocessed ASRS report. The report ID is the full ASRS report ID from the original corpus.

7.2 The Approach

After building the subjectivity lexicon, the ASRS corpus can then be analyzed to find patterns with the shaping factors and the subjective words. One such pattern that we are interested in identifying is to find a list of words for each shaping factor that are indicative of the shaping factor. That is, this list of words per shaping factor can be used as a means of wider classification of causing factors.

To do this we must first find the set of words that are present in the ASRS corpus. For each word in the trained corpus, we perform a case insensitive search to identify the list of subjective words that found within the ASRS corpus. Only words surrounded by word boundaries (punctuation and whitespace) are considered matches. That is, the word “new” will match the string “operating a new instrument panel” but will not match “he knew better”.

After the set of identified subjective words is stored into another lexicon file, we can then process this file with respect of the ASRS training data. The ASRS training data provides human labeled data that can be used to construct any number of machine learning algorithms. We are not concerned with report IDs since we are trying to aggregate knowledge about the corpus and the shaping factors.

For this project, the information gain for every word and shaping factor will be calculated. We are interested in the set of words with the highest information gain for each shaping factor as they are indicative features of the shaping factor.

The Equation 1 is modified slightly; instead of positive / negative labels, we will consider whether a shaping factor or word is absent or present in a report. That is *v* from Equation 2 will report $Values(A) \coloneqq \{present, absent\}$, where A will represent each word in the trained lexicon, and S will represent the set of reports with a shaping factor f. The *Gain(S, A)* function will be calculated for each word A and every shaping factor from [9].

7.3 Experiments and Results

For our experimentation, we operate on a list of shaping factors and a lexicon of words identified within the ASRS corpus, referred to as the ASRS Reference Lexicon (ARL). The output of the prototype implementation will report each word found in the ASRS training data, the shaping factor, and the information gain of the shaping factor given the word from the ARL.

For the first stage, we build a lexicon containing the words and phrases that are found within the ASRS corpus as described above. After searching the ASRS corpus, we were able to find the following information:

- Sentences in the ASRS corpus:1,800,594
- Unique words in the ASRS corpus:140,447
- Total words in the ASRS corpus: 32,340,425
- Unique word matches: 28,905
- Total word matches: 12,002,453

Due to the large number of lines in the ASRS corpus and the lengthy search time, the total word matches listed above only accounts for a match of a word or phrase once per line. That is, if the word “new” is used 3 times in a single sentence, it is only counted once, however, if is used in 4 different sentences, then it will be counted 4 times. The total number of unique words found equates to about 20.5807% of the total unique words used within the ASRS corpus.

The next phase in the experimentation is the reporting of the words with the highest information gain. Once the ARL is built, we loop over each word and calculate its information gain with respect to each shaping factor. Table 6 lists the top 20 words for each shaping factor identified from the training data.

Shaping Factor	Top words, sorted by Information Gain
Attitude	attitude, complacency, aviation, federal, Federal, complacent, controller, frequency, trying, pilot, mind, do, need to, need, regulation, do it, angry, too, No, no
Communication Environment	communication, environment, frequency, response, contact, controller, traffic, hear, call, told, cleared, English, radio, communications, transmission, language, maintenance, deficiency, poor, breakdown
Duty Cycle	duty, fatigue, factors, day, hour, hours, night, leg, physical, flying, fatigued, days, tired, rest, crew, 8, schedule, on duty, in a row, sleeping
Familiarity	familiarity, unfamiliar, unfamiliar with, airport, familiar, new, new to, training, familiar with, keep track of,visuals, particular, gentleman, especially, greatly, omission, stuff, concentrating, experience, time
Illusion	illusion, lights, allegedly, bargaining, black hole, blinding, breach, demarcation, exclusively, intensified, leverage, live, live with it, NB, one shot, reflecting, trap, lighting, judge, wall
Other	traffic, resolution, confusion, collision, advisory, avoidance, no other, call, alert, 2, mile, times, other than, deficiency, controller, conflict, resource, aircraft, 1, told
Physical Environment	physical, environment, turbulence, weather, visibility, moderate, overcast, the weather, rain, thunderstorm, conditions, ice, cloud, wind, scattered, low, deteriorated, severe, decision, lightning
Physical Factors	factors, physical, fatigue, duty, tired, sick, hours, fatigued, hour, rest, night, flying, leg, crew, exhausted, day, up for, disoriented, days, in a row
Preoccupation	preoccupation, distraction, distracted, attention, busy, distracting, flying, preoccupied, deficiency, quickly, duties, maintenance, resource, immediately, tuning, airspace, realized, watch, altitude, factors

Continued on the next page

Shaping Factor	Top words, sorted by Information Gain
Pressure	pressure, expedite, short time, rushed, high pressure, order, crew, checklist, old, trying, stack, test, running, in order, light, minimum, a high, with dispatch, rudder, in order to
Proficiency	proficiency, mistake, deficiency, resource, realized, error, realizing, unaware, training, reference, expect, emergency, declared, mistakes, smoke, flight attendant, oversight, performance, forgotten, unaware of
Resource Deficiency	Deficiency, resource, maintenance, engine, emergency, inoperative, smoke, gear, fire, declared, alert, proficiency, light, failure, avoidance, main, traffic, landing gear, problem, failed
Taskload	busy, controller, sector, at one, saturated, extremely, communication, overloaded, voice, traffic, too, changes, overload, deficiency, the time, manage, saturation, terminal, call, heading
Unexpected	unexpected, surprised, conducting, unusual, instinctively, causing, avoid, get behind, machine, off guard, reluctantly, startled, whatsoever, accident, opposite, made, sudden, approach, report, important

Table 7. Top 20 words per shaping factor by Information Gain

The most important observation is many of the words mirror the set of words listed by [9]. This is an indication that the subjectivity classification system is successfully capable of automatically identifying a shaping factor word list similar to the manually identified seed words.

There are a few problems with the aforementioned results. Specifically, a few words that should have been either blacklisted or marked as stop words still made it into the ARL. For example, the Unexpected shaping factor shows the word “whatsoever”, which does not appear to have a subjective sense. Another interesting result is the presence of numbers. For example, the WordNet PSC discovered and included the phrase “8” algorithms (“8#n#1”), which was shown to associate to the Duty Cycle shaping factor.

CHAPTER 8

CONCLUSIONS AND FUTURE WORK

8.1 Conclusions

The prototype system of independent subjectivity classifiers presented demonstrates that a subjectivity lexicon is a valuable resource for determining root cause analysis over a jargon heavy corpus. The prototype system employed a set of semi-supervised subjectivity classifiers to build a large and accurate lexicon. We explored the complexities surrounding the ASRS corpus and showed that the information gain statistic is an effective means of identifying a set of words most frequently associated to a shaping factor.

8.2 Future Work

Future development will focus in two primary directions: classification of the ASRS corpus and better data sources and classifiers in the pipeline subjectivity classification system to build an understanding of more jargon words.

8.2.1 ASRS – Aviation Safety Reporting System

The ASRS is a large and complex corpus. Since there is a large amount of jargon, it can be difficult to identify key features for any given report. However, this project has shown that it is still possible to mine the corpus for primary indicators of problems.

Armed with the list of words and shaping factors with the largest information gain, a traditional classifier can be built. The words that trigger high information gain for a given shaping factor could be used as features for training an SVM classifier. If a word with a high information gain for a shaping factor is located within an ASRS report, the classifier could be trained to assign the shaping factor as a label to the report. Further analysis would be needed to determine the accuracy of such as classifier. This information imposes no restrictions on any classifier that must be applied, but rather identifies key information that could be used by such a classifier.

8.2.2 Alternative Data Sources

There are other data sources that may provide data to build larger and more accurate data sources. Primarily, this project utilizes either manually annotated data sources or manually categorized data sources.

One such data source is the Wordnik website [15]. This website is a restricted access data source that provides a public API to definitions, spelling corrections, and thesaurus data. The Wordnik website has collected dictionary and thesaurus from a variety of authoritative sources written and published by humans. Currently, Wordnik supports the following dictionaries and thesauri, taken verbatim from [19]:

- The American Heritage® Dictionary of the English Language, Fourth Edition copyright © 2008, 2000 by Houghton Mifflin Harcourt Publishing Company. Updated in 2008. Published by Houghton Mifflin Harcourt Publishing Company. All rights reserved.

- The Century Dictionary
- WordNet® (WordNet 3.0 Copyright 2006 by Princeton University. All rights reserved.)
- The GNU version of The Collaborative International Dictionary of English, derived from the 1913 Webster's Revised Unabridged Dictionary
- Roget's II: The New Thesaurus, Third Edition by the Editors of the American Heritage(r) Dictionary. Copyright (c) 2003, 1995 by Houghton Mifflin Harcourt Publishing Company. Published by Houghton Mifflin Harcourt Publishing Company. All rights reserved.
- Allen's Synonyms and Antonyms, F. Sturges Allen, 1920 Harper & Brothers, New York
- User supplied wiki-style user wordlists

Due to the large amount of data, a Wordnik data source appears to be a highly valuable source of manually annotated, and authoritative, source of information. The dictionary data could be mined in the same manner as the ASL algorithm, while the thesaurus data could be mined in the same manner as WordNet.

An initial experiment showed that Wordnik is indeed a high quality data source. This theory was tested by searching for the word “pain”, which is not overtly marked as negative or identifiable by the ASL algorithm. When performing a DFS of depth 2 using only the word “pain” as the input lexicon, WordNet identifies 16 related words. In comparison, using a depth of 1, the Wordnik website will identify 155 related words, or nearly 10 times as many words and phrases. A depth of 1 for Wordnik is equivalent to a depth of 2 for WordNet queries because

WordNet queries will return multiple instances and the root of the DFS search lack the instance number, so that it may find out how many instances are available. The quality of these results has not been evaluated.

The reason this data source was not evaluated for this project is because there were complications that made training difficult. Wordnik does not have a published API access rate limit, however the server would frequently deny access based on frequency of access. There were also issues regarding the Wordnik server's reliability. The server would regularly report server errors, such as HTTP 500 (Internal Server Error) and 503 (Service Unavailable) error codes. Due to these problems, experiments and evaluation of this service could not be completed in a timely manner.

8.2.3 Alternative Pipelined Subjectivity Classifiers

Additional methods to develop subjectivity lexicons could better aid in discovery of jargon words. Of particular interest would be a semi-supervised method trained against a manually labeled data source of jargon words. It would take a considerable effort to build a lexicon of jargon words specifically tuned for the ASRS lexicon. This could be done by sifting through unlabeled phrases in the corpus and manually attaching polarity scores to selective phrases. Unsupervised learning methods similar to Hatzivassiloglou et al. could lead to improvements in jargon phrase identification.

The WordNet subjectivity classifier is currently restricted to identifying similar words and marking the entire cluster with the majority vote as described in Sections 4.5 and 5.3.4. A modification that may yield improvements would be to utilize words that have overtly marked antonyms as well. The algorithm could be modified to mark the synonyms as it currently does

and alter the polarity score of the set of antonyms in the opposite direction. That is, if the cluster of similar words indicates a positive cluster, then the antonym cluster would have its polarity score reduced because this implies it is comprise of negative words. It has not been evaluated as to the availability of antonyms, but this technique could possibly be extended to the rest of the classifiers.

REFERENCES

[1] Mohammad, Saif, Cody Dunne, and Bonnie Dorr. 2009. Generating High-Coverage Semantic Orientation Lexicons From Overtly Marked Words and a Thesaurus. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 599-608, Singapore.

[2] Esuli, Andrea and Fabrizio Sebastiani. 2006. SentiWordNet: A publicly available lexical resource for opinion mining. In *Proceedings of LREC*, pages 417-422.

[3] Turney, Peter and Michael Littman. 2003. Measuring Praise and Criticism: Inference of Semantic Orientation from Association. In *ACM Transactions on Information Systems (TOIS), 21*(4), pages 315-346.

[4] Suzuki, Yasuhiro, Hiroya Takamura, and Manabu Okumura. 2006. Application of Semi-supervised Learning to Evaluative Expression Classification. In *Lecture Notes in Computer Science 3878*, pages 502-513.

[5] Hassan, Ahmed and Dragomir Radev. 2010. Identifying Text Polarity Using Random Walks. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 395-403, Uppsala, Sweden.

[6] Wilson, Theresa, Janyce Wiebe, and Paul Hoffman. 2005. Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis. In *Proceedings of HLT-EMNLP-2005*.

[7] Sokolova, Marina, Nathalie Japkowicz, and Stan Szpakowicz. 2006. Beyond Accuracy, F-score and ROC: a Family of Discriminate Measures for Performance Evaluation. In *Lecture Notes in Artificial Intelligence* 4304, pages 1015-1021.

[8] Mandala, Rila, Takenobu Tokunaga, and Hozumi Tanaka. 1999. Complementing WordNet with Roget's and Corpus-based Thesauri for Information Retrieval. In *Proceedings of the ninth conference EACL*, pages 94-101.

[9] Abedin, Muhammad Arshad Ul, Vincent Ng, and Latifur Khan. 2010. Cause Identification from Aviation Safety Incident Reports via Weakly Supervised Semantic Lexicon Construction. In *Journal of Artificial Intelligence Research 38*, pages 569-631.

[10] Gün. Semin and Klaus Fiedler. 1988. Cognitive Functions of Linguistic Categories In Describing Persons: Social Cognition and Language. In *Journal of Personality and Social Psychology*, 54, pages 558-568.

[11] General Inquirer. http://www.webuse.umd.edu:9090 (accessed 22 August 2010).

[12] General Inquirer – Pos Tagged Words. http://www.webuse.umd.edu:9090/tags/TAGPos.html (accessed 22 August 2010).

[13] General Inquirer – Neg Tagged Words. http://www.webuse.umd.edu:9090/tags/TAGNeg.html (accessed 22 August 2010).

[14] General Inquirer – Marker Categories. http://www.wjh.harvard.edu/~inquirer/kellystone.htm (accessed 14 November 2010).

[15] Moby Project. http://icon.shef.ac.uk/Moby (accessed 15 July 2010).

[16] WordNet. http://wordnet.princeton.edu (accessed 15 July 2010).

[17] Artstein, Ron and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics (survey article). In *Computational Linguistics 34(4)*, pages 555-596.

[18] Wordnik. http://www.wordnik.com (accessed 15 October 2010).

[19] Wordnik FAQ. http://www.wordnik.com/faq (accessed 15 October 2010).

[20] Sandhaus, Evan. 2008. The New York Times Annotated Corpus. Linguistic Data Consortium, Philadelphia.

[21] Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. 2009. *Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. Springer Series in Statistics. New York: Springer Science+Business Media, LLC.

[22] Baccianella, Stefano, Andrea Esuli, and Fabrizio Sebastiani. 2010. SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*. European Language Resources Association (ELRA).

[23] Russell, Stuart J., Peter Norvig. 2006. *Artificial Intelligence: A Modern Approach*. 2nd ed. New Jersey: Pearson Education, Inc. (orig. pub. 2003.)

[24] Mitchell, Tom M. 1997. *Machine Learning*. N.p.: WCB/McGraw-Hill.

[25] Lewis, D. D. RCV1-v2/LYRL2004: The LYRL2004 Distribution of the RCV1-v2 Text Categorization Test Collection (14-Oct-2005 Version). http://www.jmlr.org/papers/volume5/lewis04a/lyrl2004_rcv1v2_README.htm.

[26] ASRS Stop Words. http://www.github.com/s1n/thesis/tree/master/data/smartstop.txt (created 17 November 2010).

[27] Quinlan, J. R. 1986. Induction of Decision Trees. Machine. Learning 1, 1 (Mar. 1986), pages 81-106

[28] Fellbaum, Christiane. 1998, ed. WordNet: An Electronic Lexical Database. Cambridge. MA: MIT Press.

[29] General Inquirer. http://www.wjh.harvard.edu/~inquirer/ (accessed 22 August 2010).

[30] General Inquirer – Note Introducing Server Version of General Inquirer. http://www.wjh.harvard.edu/~inquirer/server_blognote.html (accessed 14 November 2010).

[31] Wall, Larry, Tom Christiansen, and Jon Orwant. 2000. *Programming Perl.* 3rd ed. Sebastopol, CA: O'Reilly & Associates, Inc.

[32] Friedman, Jerome, Trevor Hastie, and Robert Tibshirani. 2000. Additive logistic regression: A statistical view of boosting. In *The Annals of Statistics*, Vol. 28 No. 2, pages 337-407.

[33] Riloff, E. and J. Wiebe. 2003. Learning Extraction Patterns for Subjective Expressions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2003)*, pages 105-112. Sapporo, Japan.

[34] Posse, C., Matzke, B., Anderson, C., A., Matzke, M., & Ferryman, T. 2005. Extracting information from narratives: An application to aviation safety reports. In *Proceedings of the 2005 IEEE Aerospace Converence*, pages 3678-3690.

[35] Artstein, R., and Poesio, M. 2008. Inter-coder agreement for computational linguistics. In *Computational Linguistics, 34*(4), pages 555-596.

[36] Jané, Jesús Llevadias, Stephen Helmreich, and David Farwell. 2005. Identifying Jargon in Texts. In *Procesamiento del Lenguaje Natural*, no. 35, pages 425-432.

[37] Dictionary facts – Oxford English Dictionary. http://oed.com/about/facts.html

[38] Reynard, W. D., C. E. Billings, E. S. Cheaney, and R. Hardy. 1986. The Development of the NASA Aviation Safety Reporting System. In *NASA Reference Publication 1114*. California: NASA Scientific and Technical Information Branch.

[39] Kullback, S. and Leibler, R. A. 1951. On Information and Sufficiency. In *Annals of Mathematical Statistics 22*(1). pages 79-86.

[40] Alessio Damato and Brona. 2007. Binary Entropy Plot. From Wikipedia website. http://en.wikipedia.org/wiki/File:Binary_entropy_plot.svg (accessed 24 November 2010).

APPENDIX A

Below is a complete list of Affix Seed Subjectivity Pairing (ASSP) regular expression patterns used to build the Affix Seed Lexicon (ASL):

Unmarked / Positive Pattern	**Marked / Negative Pattern**	**Example Pairing**
^(.+)$	dis$1	honest, dishonest
^(.+)$	im$1	possible, impossible
^(.+)$	in$1	consistent, inconsistent
^(.+)$	mal$1	adroit, maladroit
^(.+)$	mis$1	fortune, misfortune
^(.+)$	non$1	sense, nonsense
^(.+)$	un$1	happy, unhappy
^(.+)$	$1less	gut, gutless
^(l(.+))$	il$1	legal, illegal
^(r(.+))$	ir$1	responsible, irresponsible
^(.+)ful$	$1less	harmless, harmful

where "$1" is the matched pattern from the unmarked/positive half of the ASSP. The remaining pattern characters are valid Perl Regular expressions [31].

APPENDIX B

The following is a list of stop words removed from [25]:

able	against	allow	always	appear	appreciate	appropriate
associated	available	awfully	became	become	becomes	becoming
believe	best	better	brief	cause	causes	certain
certainly	changes	clearly	consequently	consider	considering	contain
containing	contains	corresponding	course	currently	definitely	described
despite	different	do	does	doesn't	doing	done
enough	entirely	especially	even	exactly	example	except
far	followed	follows	former	formerly	further	get
gets	getting	greetings	happens	hardly	hello	help
here	hi	hopefully	ignored	immediate	inc	indeed
indicate	indicated	indicates	inner	instead	inward	just
know	knows	known	lately	latter	latterly	least
less	let	like	liked	likely	little	may
mean	might	more	must	name	necessary	need
needs	never	new	no	novel	often	old
ought	out	outside	overall	own	particular	particularly
placed	please	possible	probably	provides	reasonably	regardless
right	said	same	saw	say	saying	says
second	secondly	see	seeing	seem	seemed	seeming
seems	seed	sensible	sent	serious	seriously	soon
sorry	specified	specify	specifying	still	sub	sup
sure	take	taken	tell	tends	thank	thanks
thanx	think	torough	thoroughly	too	took	tried
tries	truly	try	trying	unfortunately	use	used
useful	uses	using	usually	value	various	welcome
well	went	whither	whole	willing	wish	wonder
yes						

The following is a list of blacklisted words added to [25]:

alongside	atop	being
each other	thee	thyself
till		

The list of remaining words in [26] were used during training.

VITA

Jason Switzer was born in Austin, Texas on May 5, 1982, the son of Paul and Patricia Switzer. After graduating with Honors from Round Rock High School, Round Rock, Texas in 2000, he entered the University of Texas at San Antonio. In July 2003, he took a software development position at Secorp Technologies, where he worked full-time while pursuing his degree full-time. He received his Bachelor of Science in August 2004, majoring in Computer Science. In May 2005, he took a position as a Software Engineer at L-3 Communications working in the Special Systems group developing the state of the art Human-Computer Interface systems. In December 2010, he will receive his Masters of Science in Computer Science in the field of Intelligent Systems. In May 2011, he will become a father to his first-born child.