USING FREEBASE, AN AUTOMATICALLY GENERATED DICTIONARY, AND A CLASSIFIER TO IDENTIFY A PERSON'S PROFESSION IN TWEETS

by

ABRAHAM HALL
B.S. Brigham Young University, 2005

A thesis submitted in partial fulfillment of the requirements
for the degree of Master of Science
in the Department of Electrical Engineering and Computer Science
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Summer Term
2013

Major Professor: Fernando Gomez

ABSTRACT

Algorithms for classifying pre-tagged person entities in tweets into one of eight profession categories are presented. A classifier using a semi-supervised learning algorithm that takes into consideration the local context surrounding the entity in the tweet, hash tag information, and topic signature scores is described. In addition to the classifier, this research investigates two dictionaries containing the professions of persons. These two dictionaries are used in their own classification algorithms which are independent of the classifier. The method for creating the first dictionary dynamically from the web and the algorithm that accesses this dictionary to classify a person into one of the eight profession categories are explained next. The second dictionary is freebase, an openly available online database that is maintained by its online community. The algorithm that uses freebase for classifying a person into one of the eight professions is described. The results also show that classifications made using the automated constructed dictionary, freebase, or the classifier are all moderately successful. The results also show that classifications made with the automated constructed person dictionary are slightly more accurate than classifications made using freebase. Various hybrid methods, combining the classifier and the two dictionaries are also explained. The results of those hybrid methods show significant improvement over any of the individual methods.

ACKNOWLEDGMENTS

I would like to thank my thesis committee members Fernando Gomez, Damian Dechev, and Marshall Tappen for their support and help in the research I have been doing for this work. Also, I would like to thank Mosaic ATM for their financial support the past five years as I have worked on my graduate degree.

Lastly, I would like to thank my wife, who has shown remarkable patience with me and has been a constant source of encouragement throughout the entirety of my coursework and thesis work.

TABLE OF CONTENTS

LIST OF TABLES

CHAPTER ONE: INTRODUCTION

Twitter has become an increasingly popular form of communication on the internet and potentially can offer a wealth of information in the field of Natural Language Processing (NLP). Users often tweet multiple times a day, meaning that tweets can offer much more up to the minute information than traditional mediums like newspapers or magazines. Additionally, the total amount of information potentially available from Twitter is staggering -- about one billion tweets per week (Twitter Blog). For these reasons, more and more work is being done to produce good NLP tools for use with Twitter and other social media sites.

However, dealing with the informal language often contained in tweets offers challenges for working with the text. Language in tweets is often informal, containing abbreviations, improper capitalization, misspellings (sometimes intentional), and poor punctuation (Ritter, Clark and Etzioni). These are all things that make it difficult to use out of the box natural language tools that have been trained on traditional, more formal corpora such as newspaper text. These tools often rely on things like punctuation and capitalization in order to accurately recognize named entities or parts of speech. To further complicate things, tweets are limited to 140 characters, which can make it difficult to get enough context from the tweet to properly interpret its meaning.

Named Entity Recognition (NER) is one area where widely used tools do not translate well to tweets, although recent work in this area has shown an increased ability to accurately tag entities in tweets (Li, Weng and He), but only at a coarse grained level. Examples of coarse grained

entity categories are Person, Location, or Organization. In this work, however, the main interest is in assigning fine grained classifications. So the focus is on assigning an already tagged Person into one of eight subcategories representing classes of professions. This information can later be used in Question/Answering systems or to apply filters to the vast amount of information in the available Twitter stream.

It should be noted that the sub-categorization problem is inherently more difficult than the coarse grained classification problem, even for humans (Fleischman). For instance, in the following tweets, taken from the corpus, it is relatively easy to determine that the blanked out portion of the tweets most likely refer to Persons.

1. My views on migraines, as expressed by_____.
2. I read all of your tweets in _____'s voice
3. _____ Tells the Gun Industry why President Obama is the Best Friend They've Ever Had.

But trying to determine the profession of the person from the above tweets is far more difficult. Based on the context, one could easily assume the person in the first example to be a doctor. The second tweet is sufficiently vague enough that it could refer to anyone. And the person in the third tweet seems to be a politician of some sort. However, the person in the first tweet is Colin Firch, an actor; the person in the second tweet is Patrick Dempsey, also an actor; and the person in the third tweet is Bill Maher, a comedian. This illustrates the difficulty in assigning finer grained classifications to entities, especially in tweets where the context is oftentimes limited.

In order to properly make these assignments, three strategies were investigated. The first strategy was a semi-supervised learning method adapted from research done on fine grained classification of person entities (Fleischman and Hovy). A summary of their method and algorithm will be presented, as well as an explanation of additional features that were investigated as part of this research. The results show that these additional features improve the classifier's performance on tweet data.

The second strategy is an unsupervised method that used the Web to dynamically build a resource that contains person/profession relationships, referred to as a person dictionary. The dictionary was then used to make the profession classification. The algorithm for automatically acquiring data to build the person dictionary will be described in this paper. The metric for choosing a profession category when dealing with ambiguous entries in the dictionary will also be discussed.

The third strategy is also an unsupervised method that used a readily available online database called freebase (Bollacker, Evans and Paritosh). In this method, the entity was looked up in the freebase database, and the search results were used to make the profession classification. Several methods of choosing the best search result and the best profession classification are explored in this paper.

Lastly, several hybrid methods were investigated which use the different methods in conjunction with each other to achieve better results. These methods attempt to use the unique strengths of

each of the individual methods in order to produce a more accurate classification system. These methods will be explained later in the paper as well.

CHAPTER TWO: RELATED WORK

At a high level, there are two possible approaches for assigning a fine grained classification to an entity. The first approach is to train the recognition tool in more than just the basic entity categories. Some NER tools have been developed that can accurately assign entities in as many as a hundred different categories (D. Nadeau). However, as stated before, these tools, which are trained on formal corpora such as newspaper articles, are not nearly as effective on tweets because of the generally low quality of tweet text (Locke and Martin). Some recent work has been done with the goal of improving not only NER tools but general NLP tools for tweets as well. For example, a Part-of-Speech (POS) tagger developed specifically for Twitter has been shown to be an improvement over the out of the box Stanford tagger (Gimpel, Schneider and O'Connor). This is notable as POS tagging is often the foundation upon which other NLP research is built (Toutanava, Klein and Manning).

The second strategy is to tag data at a coarse grained level, and then to build a separate system to do the sub-classification (Fleischman and Hovy). This is the strategy pursued in this work as it is already a challenging problem to assign coarse grained categories to named entities in Twitter without the added complexity of adding in fine grained categories as well. Also, since so much work recently has focused on improving the coarse grained classification of entities in Twitter and other social media sites, it seemed prudent to focus this work on making finer grained classifications from coarse grained classifications in order to maximize the opportunity to make a meaningful contribution.

Despite the difficulties of using tweets, it has been shown that by improving the quality of the text in tweets, the NER classification performance tools can be improved. Some such methods of doing this are to correct misspellings, to replace abbreviations, and to determine if capitalization in the tweet is meaningful or not (Ritter, Clark and Etzioni). Similarly, work has been done to improve the quality of text messages and tweets to improve NER classifications using a classifier that identifies ill-formed words (typos, abbreviations, emoticons, etc.) and replaces them with proper words (Han and Baldwin). A study has shown that in tweets, a very small number of terms make up very large proportion of these types of improper words (Gouws, Metzler and Cai) which further supports the assertion that pre-processing the text in tweets is a relatively simple way to greatly improve both the quality of text and the resulting classifications.

Several proposed methods for making NER classifications in tweets have been presented recently. A method that combines a K-nearest neighbors classifier with a Conditional Random Fields model has also shown increased accuracy over traditional NER tools when dealing with tweets (Liu, Zhang and Wei) Other work has shown that using context data from Wikipedia and the Web N-Gram corpus can improve a classifier's ability to identify entities, although in this work, it did not attempt to classify the entity into one of the three basic coarse grained categories of Person, Location, or Organization. Rather, it only looked at the ability to find a named entity (Li, Weng and He). And yet another paper looked at the efficiency and accuracy of using Amazon Mechanical Turk and CrowdFlower as an option for tagging named entities in Twitter (Finin, Murnane and Karandikar).

Additional work has shown that using available resources such as OntoNotes has proven to be helpful when doing NLP work (Schwartz, Gomez and Ungar). Similarly, a system which uses Wikipedia as an outside source of information in NER work has shown promise as well (Cucerzan). Several people have attempted to use Wikipedia to automatically produce reference resources as well. One such work attempts to use Wikipedia to produce a corpus that would be available to train taggers for NER (Nothman). Another work attempts to use Wikipedia to create a two hundred entity named entity dictionary (Higashinaka, Sadamitsu and Saito). While yet another produces a reference gazetteer by using WordNet that can then be used to more accurately identify people in tweets (Maynard and Funk). All of this work is similar in some ways to the work presented in this paper, in which one of the goals is to automatically generate a person/profession dictionary for use in assigning named entities to a profession class.

There has also been some work done that attempts to use freebase as an alternate resource for performing NLP tasks. One such project uses the celebrity and video game lists from freebase in an attempt to better identify proper nouns in tweets (Owoputi, O'Connor and Dyer). And in a different work, freebase and other readily available databases are used to create a better natural language question and answering system (Yahya, Berberich and Elbassuoni).

.

CHAPTER THREE: METHODOLOGY

This classification system takes a tweet that has already been tagged with the basic coarse grained classifications of Person, Organization, and Location. It then attempts to assign the person(s) into one of eight profession categories: Athlete, Businessperson, Doctors, Entertainers, Journalists, Politicians, Clergy, and Authors. These categories were chosen because of their relative frequency in the corpus versus other categories. As mentioned in the Introduction, three distinct methods were investigated for making these classifications, along with several hybrid methods that used parts of each algorithm. These methods were a classifier, a person/profession dictionary, and a freebase lookup.

For the classifier, a bootstrapping method was used to generate a training set upon which the classifier was then trained. The classifier was then run over the tweet corpus and any classification with a confidence level greater than a configurable threshold was saved in a table. Through experimentation, the ideal value for this threshold was found to be 82.5%. Next, any additional instances of that same person in the test set were then given the same profession classification as the most frequent entry for that person in the table. This was based on the assumption that subjects in Twitter often trend, so tweets collected in the same time frame that mention the same name are most likely referencing the same person.

The person dictionary method uses a reference file containing person/profession relations to classify each person. For this method an automatically generated dictionary was used for the classifications and the results compared against classifications made with a manually compiled

dictionary. For each person in each tweet, a web search (Yahoo! BOSS Search) on the person was run and the potential professions were then extracted from the query results. These person/profession relationships were then saved to a reference file that was used to classify each person into one of the eight categories. For persons with multiple profession entries in the file, a disambiguation metric was applied and the classification that received the best score was then chosen as the classification. If there were no profession entries in the dictionary for the person, then the person was assigned the default profession of Entertainer, the most common profession in the corpus.

In the freebase method, a freebase query was run for each person. Each query generally returned multiple hits, so various methods for determining the best match freebase entry were explored. Once the best match freebase entry was determined, the entry was used to determine the best profession classification for the person. If no profession classification was found in the freebase entry, then the person was assigned the default profession of Entertainer.

Lastly, various hybrid methods using the classifier, the person dictionary, and freebase were explored and found to have varying degrees of accuracy, though most all of them outperformed each of the methods individually. All of this will be covered in greater detail in the rest of the paper.

Data Set Generation

The corpus consists of a set of roughly 7 million tweets collected via the sample interface provided by the twitter API over a time period beginning on March 14, 2012 and ending on April

10, 2012. Duplicate tweets were eliminated by use of the tweet ids and non-English tweets were removed by using the language tags available in the raw data. Retweets were also eliminated to ensure that there was not a lot of repetition within the set. Next the tweets were stemmed (Porter), tokenized (Apache Open NLP), and entity tagged (The Stanford Natural Language Processing Group). This resulted in a final corpus of about 1.2 million tweets that contained a tagged person entity.

The tweets were then preprocessed to make the text easier to deal with. A list of common abbreviations was manually compiled, and used to replace the abbreviations in the corpus with the correct English words. Contractions were also replaced with the full English equivalents. Additionally, a misspellings dictionary was put together that was used to correct misspellings in the corpus. Usernames were removed from the text as well. In Twitter, usernames are always preceded by an '@' so they were easy to identify and remove. Lastly, any html style tags were removed as well.

From the preprocessed corpus, a hundred seed instances were manually annotated for each of the eight profession categories, creating an initial training set of eight hundred instances. Next, using lists found on the internet, a person dictionary of about ten thousand entities was manually compiled to be used in the bootstrapping method (Fleischman and Hovy) that was used to generate the training data. The classifier, which will be described in more detail later, was trained on the seed sets, and then used to classify the remaining corpus. Any tweet that had an entity with a classification confidence greater than a configurable threshold that also matched a person/profession relation found in the manually compiled dictionary was then added to the

training set. Any tweet with a profession classification with a confidence level above the threshold that didn't match an entry in the person dictionary was written out to a separate file and then manually verified before being added to the training set. This process was repeated over several iterations until a training set of just over thirteen thousand tweets was produced.

For testing the system, two separate testing sets were created. The first set was comprised of one thousand randomly selected tweets from the training set. These tweets were removed from the final training set and saved in a separate set referred to throughout the paper as the Validation Set. This resulted in the final training set ultimately containing about twelve thousand tweets. A second set was compiled from tweets not included in the training set and was used to test the generalizability of the classification system. This separate set is referred to as the Hold Out Set throughout the document and contained a little more than thirteen hundred tweets.

Describing the Classifier

The classifier used by this system was a C4.5 classifier (Quinlan). The classifier was trained on and used several types of features to make the profession classifications. The two main types of features used to train this system were word frequency features (Fleischman) and topic signature scores (Fleischman and Hovy). Additionally, hash tags were used by weighing them in the topic signature score calculation. Also, pattern matching was used to generalize a few types of data in the tweets to improve the word frequency and topic signature score features. Lastly, a table was used to store all the high confidence classification results for each entity. This table was then used to change lower confidence classifications for matching entities (Fleischman).

Word Frequency Features

Word frequency features (Fleischman) measure the frequency of words directly before and after each person instance in a tweet. The idea behind this is that words or combinations of words that appear in close proximity to the person in the text are more likely to be indicators of what profession classification should be made for that person. For each instance of a person in a tweet, ten word n-grams were tracked in the word frequency table. They are the three unigrams directly preceding the person, the three unigrams directly following the person, the bigram directly preceding the person, the bigram directly following the person, the trigram directly preceding the person, and the trigram directly following the person. For example, in the tweet:

> "I just wanted to clarify that I was in love with Josh Hutcherson when he was in Zathura and Bridge to Terabithia."

The n-gram features for "Josh Hutcherson" would be "in", "love", "with", "when", "he", "was", "love with", "when he", "was in love" and "when he was". Each of these n-grams would then be tracked in the word frequency table for the type Entertainer, which is Josh Hutcherson's correct profession category.

Table 1
An Excerpt from the Word Frequency Table

N-Gram Type	N-Gram Word(s)	Profession	Frequency Count
Following Unigram	Score	ATHLETE	29
Previous Unigram	Off	ATHLETE	9
Previous Unigram	Off	POLITICIAN	0
Previous Unigram	Off	BUSINESS	1
Previous Unigram	Off	ENTERTAINER	6

A word frequency table was created for the training set and saved to be used in the classification process. For each person instance to be classified, the n-grams were found and then the frequency counts for each n-gram were looked up in this table. The count for each profession category and each n-gram was then used as a feature for the classifier, producing a total of eighty features per instance.

Topic Signature Score Features

Topic signature scores (Lin and Hovy) were also used as a feature in this classifier. The purpose of topic signature scores is to find terms that act as strong indicators that an entity should be classified with a certain profession category. For each unigram in the word frequency table and for each profession category, a likelihood ratio between two hypotheses, referred to as the λ-score, was calculated (Dunning). The hypotheses were made using a couple of different probability calculations. The first probability (p1) was a measure of the probability that, given a specific unigram, the entity in question was a member of a specific profession category. The second probability (p2) was a measure of the probability that, given any other unigram, the entity was a member of the profession category. The first hypothesis (h1) then stated that the first probability was equal to the second probability (h1: p1 = p2). The second hypothesis (h2) stated that the second probability was much larger than the first probability (h2: p2 >> p1). This λ-score was then calculated using these hypotheses, as defined in Equation 3.1.

$$\lambda - \text{score} = -2\log\frac{L(h1)}{L(h2)} \tag{3.1}$$

To calculate the likelihood for each hypothesis, a contingency table was created that tracked the number of times the unigram term was a positive indicator for the profession category in question, the number of times the unigram term did not indicate that the entity is a member of the profession category in question, the number of times any other unigram term was an indicator for the profession category in question, and the number of times any other unigram term was not an indicator for the profession category in question

Table 2
The Contingency Table used in the Likelihood Calculations

	R	R'
t_i	O_{11}	O_{12}
t_i'	O_{21}	O_{22}

R denotes the entity is in the relevant profession category.
R' denotes that the entity is not in the relevant profession category.
t_i is the unigram term in question.
t_i' indicates every other unigram in the table.
O_{11} represents the frequency that the unigram term was an indicator that the entity was in *R*.
$O_{12}$is the frequency the unigram term does not indicate the term is in *R*.
O_{21} is the frequency that any other unigram indicates the entity is in *R*.
O_{22} is the frequency that any other term indicates the entity is not in *R*.

Using the contingency table, the ratio of the likelihood of the two hypotheses as expressed in Equation 3.1 is calculated. The equation for calculating the value of this ratio, using the contingency table values is expressed in Equation 3.2.

$$\frac{L(h1)}{L(h2)} = 2 \sum_{i,j} O_{i,j} \, log \frac{n_{i,j}}{m_{i,j}} \tag{3.2}$$

Equations 3.3 and 3.4 further define the terms $n_{i,j}$ and $m_{i,j}$ respectively.

$$n_{i,j} = \frac{o_{i,j}}{o_{i,1} + o_{i,2}} \tag{3.3}$$

$$m_{i,j} = \frac{o_{1,j} + o_{2,j}}{o_{11} + o_{12} + o_{21} + o_{22}} \tag{3.4}$$

Once the λ-score had been calculated, terms that were not strong indicators of an entity's membership in a specific profession category were filtered out. This was done by calculating the ratio P (R| t_i)/ P(R/ t_i'). If this ratio ended up being greater than the mean ratio for every other unigram term for the profession category, it was then stored in a table containing all the relevant terms and their λ-score.

Table 3
An Excerpt from the λ-score Table

Unigram	Category	λ-score
profession	BUSINESS	5.932654074112653
speaker	POLITICIAN	3.621740646577735
speaker	BUSINESS	5.932654074112653
pioneer	DOCTOR	16.050134909921784
charity	BUSINESS	11.868110103909721

Once the table was compiled, the topic signature score was calculated for each person instance to be classified for each profession category, as described in Equation 3.5, using the number of words in the tweet as N.

$$\sum_N \frac{\lambda\text{−score of } word_{n,type}}{(distance\ from\ entity)^2} \tag{3.5}$$

It should be noted that the topic signature score weights the λ-score according to how closely it appears in the text to the entity, assuming that the closer to the entity a word appeared, the more relevant it was. This was not always a correct assumption, but as a general rule it worked well. Each of the calculated topic signatures scores was then used as a feature for the classifier, resulting in a total of eighty eight features.

Hashtag Weights

Hashtags are unique to Twitter and provide a way for the tweeter to indicate a topic or topics that are relevant to the tweet. This kind of manual categorization is a very useful way that context information can be gained from the tweet. A hashtag is denoted with the character '#' and is easily extracted from the text. Or alternately, the raw data obtained through the twitter API contains an entry denoting all the hash tags used, so the hash tag information can be taken from there as well.

For this classifier, the hashtag frequencies were tracked in a frequency table similar to the word frequency table. The major difference was that while the word frequency table tracked only n-gram combinations that appeared in proximity to an entity in the tweet, the hashtag frequency table tracked all hashtags that appeared in a tweet with an entity.

These frequencies were then used to calculate λ-scores for each of the hashtags in the same manner as the λ-scores for each unigram was calculated. The topic signature score calculation was then modified to always use the full weight of the hashtag λ-score. In other words,

regardless of where the hashtag appeared in relation to the entity in the tweet, the full λ-score was used in the topic signature score.

Using Pattern Matching to Generalize Data

One issue with the word frequency features used by the classifier was that the n-grams were stored precisely as they appeared in the tweet text. This led to problems in certain cases some n-grams that intuitively should have been indicative of a specific profession category were not. For example, in the following tweet, the term "88-85" would have been saved in the word frequency table as one of the n-grams.

> "Lakers beat Hornets 88-85 on Bryant's late 3 (Yahoo! Sports) : Kobe Bryant hit a go-ahead 3-pointer with 20 secon…"

Similarly, in the next tweet, the term "98-97" would have been saved in the word frequency table as an n-gram as well.

> "#Sports&HighLights!! Clippers Hold Off Blazers 98-97: Chris Paul, Blake Griffin and Randy Foy... http://t.co/Kvb0KEgJ Keep it Locked!!!"

In these tweets, the terms that indicate the score would have been tracked and counted as unique n-grams in the word frequency table, but a ballgame score of any sort would seem to be an indication that the entity in the tweet was most likely an athlete. In order to improve the word frequency features, pattern matching (Hearst) was used to consolidate n-grams that represented the same thing under a single tag. Any text matching a pattern was replaced in the text by a tag

indicating the type of pattern it matched, and the resulting entry in the word frequency table was tracked using the pattern tag.

So for the tweets in the above examples, the terms “88-85” and “98-97” matched score patterns and were replaced by <SCORE> tags. This resulted in both tweets incrementing counts for <SCORE> in the word frequency table, and in general, improved the word frequency scores.

In addition to using the patterns to improve the word frequency scores, the pattern matches were also tracked in a separate frequency table similar to the hashtag table. The idea here was that again, something like a score would be an indicator of an athlete regardless of where it appeared in the tweet. So for any pattern match in any tweet containing an entity, the match was tracked in this frequency table regardless of its proximity to the entity.

This pattern tag frequency table was then used to calculate λ-scores and those λ-scores were then given full weight in the topic signature scores, just like the hashtag λ-scores were. Several different patterns were tried including url, phone numbers, dates, and time, but ultimately, only three patterns were found to have a significant effect on the results. Those patterns were scores, which were usually game scores or vote totals; ratings, which were given in reviews often in an X out of Y format; and references, which were generally citations of bible verses.

Using the High Confidence Classification Table

On Twitter, topics often trend, meaning that a lot of users are tweeting about the same topic at around the same time. Because of this trending behavior, tweets that mention people who have

the same name are most likely referencing the same person. For example, the corpus collected for this project took place during the men's and women's NCAA basketball tournaments. As such, any mention of "Anthony Davis" was almost certainly a reference to the University of Kentucky basketball player Anthony Davis, who was named Most Outstanding Player at that tournament, and not to offensive tackle Anthony Davis of the San Francisco 49ers or Anthony Davis, American composer.

In order to take advantage of this quality, classifications in which the classifier had a high degree of confidence were stored internally by the classifier in a table (Fleischman). The table tracked the average confidence value for an entity and a specific profession, as well as the number of high confidence classifications made by the classifier for the entity/profession combination.

Table 4
An Excerpt from the High Confidence Classification Table

Person	Profession	Occurrence Count	Confidence Level
Josh Hutcherson	ENTERTAINER	1	0.9396551847457886
Kobe Bryant	ATHLETE	2	0.9078947305679321
Katy Perry	ENTERTAINER	4	0.9149686845133591
Katy Perry	AUTHOR	1	0.8333333134655184

Each time an entity was classified, a look up to this table was made, and if a match was found, the classifier's classification was replaced with the best match classification from the table. In cases where an entity had multiple profession categories in the table, the best match was determined by taking the number of classification matches multiplied by the average confidence

value and the profession category that had the highest score was returned as the best match for the entity.

In addition to taking advantage of trending topics in Twitter, this table also helps the system to evolve over time. If over time a person with the same name as another person but with a different profession starts to trend, eventually the number of entries for the person with the new profession will overtake the old profession as the best match in this table, resulting in classifications of the new profession category for that entity moving forward. The same is true if the same person starts to trend, but for a different profession. So if Mitt Romney, who was running for president during the corpus collection, were to suddenly start trending on Twitter again for something related to his current work as a businessman, eventually the number of business entries in the table would outweigh the number of politician entries in the table and Mitt Romney would be classified as a businessman from then on.

Using the Web to Classify an Entity

One of the biggest drawbacks to the classifier is that it relies on the content of the tweet, especially in close proximity to the entity. But in the following tweet example, there is no content that the classifier can really use:

> "Justin Bieber!!!"

But even in the best of tweets, because of the limitation of 140 characters imposed on them, it can be difficult to get enough information from the tweet to properly make a classification. For that reason, an alternate method of making classifications was developed. For this method, a call

to an internet search API was made in an attempt to get more usable data on an entity and to then use that information to make a classification. The resulting entity/classification pair was then saved in a reference file referred to a person dictionary and used to make a profession classification.

Recall that in the bootstrapping method that was used to generate the training data for the classifier, a person dictionary was manually compiled and was then used to verify high confidence classifications before adding the tweet to the training corpus. Using the manual dictionary as a reference file in the person dictionary method showed promise, and so the time was spent to create a method whereby a person dictionary could be automatically generated using the web. This eliminated the need for the time consuming process of manually compiling such a file.

For this method, an empty dictionary was created and then as web searches were performed for each entity that was classified, the resulting person/profession pairs were saved to this file. The web searches were only performed for entities that did not already have an entry in the dictionary to cut down on the number of web queries. For any entity that did not find a profession match via the web search or the dictionary, the default value of Entertainer was assigned, as it was the most common profession in the data sets. In cases where the web search results or the dictionary contained multiple potential professions for an entity, a disambiguation metric was applied and the category that returned the highest score was used as the classification for the entity. This is all explained in further detail in the remainder of this section.

Automatically Generating the Person Dictionary

For each person entity to be classified, a web query was run on the person's name (Yahoo! BOSS Search). Next a list of patterns was compiled to be used to extract the person's profession from the web results. Examples of these patterns include "<name> is a", "<name> was a", "<name> became a", and "<name> served as a". Additional patterns were created that ignored data that might typically be found in a biographical entry for a person such as birth date, birth place, and death date. Additionally, patterns which ignored prefixes, suffixes, and middle names were also created in order to maximize the number of pattern match hits the system found. Lastly, a nicknames dictionary was compiled that contained a list of common nicknames such as "Lucy" in place of "Lucille". Using the nickname dictionary allowed for more pattern matches in cases where the tweet contained the nickname and the web result contained the full name, or vice versa.

During the processing of the web query results, only the first sentence of the result was checked for pattern matching. The idea behind this was that in a biographical entry, the first sentence is likely the one that contains the most relevant profession information. Other things the person may be less known for will generally show up after the first sentence. After the first sentence of a result was extracted, it was then POS tagged and all nouns following the pattern match were identified. These nouns were then checked against an occupations file that used an IS-A relationship to define occupation/profession category relations. For example, a signer is an entertainer, so a look up of singer in this file would return that the correct profession category was Entertainer. An entry was then saved in the person dictionary for each entity/profession pair found during the web query result processing.

Dealing With Ambiguity in the Web Search Results and Dictionary

In cases where an entity had only one matching profession in the dictionary, making the classification was easy. But in cases where the dictionary contained multiple entries for a person, a disambiguation metric had to be applied in order to make a classification.

There were two ways ambiguity could be introduced into the dictionary. The first way was when a single person was known for multiple professions. For example, Michael Jordan is best known as an athlete. But Michael Jordan is also a businessman who owns the Charlotte Bobcats. And he also is the star of the film Space Jam. Thus, the dictionary could end up with three potential professions for Michael Jordan. But most people would classify Michael Jordan as an athlete, as that is what he is best known for, so for the purposes of creating the truth data, an athlete classification for Michael Jordan was considered the correct one. In general, for all persons who were known for multiple professions, the most notable profession for the person was treated as the correct classification.

The second way ambiguity could be introduced into the person dictionary was when different people who were known for different professions shared the same name. For example, Michael Lewis is a retired free safety who played in the NFL. But there is another Michael Lewis who is an author, another who is a journalist, and yet another who is a saxophonist. In these cases, the correct classification is difficult to determine unless there is something in the content of the tweet that makes it obvious.

The disambiguation metric used to determine the correct classification for an entity was based on the Resnik work similarity score (Resnik) between each word in the tweet and each potential profession, omitting all proper nouns as shown in Equation 3.6. In this equation, the *srm* function is the Resnik similarity score. Also, c_t, c_c represent a sense of the target word (w_t) and the category word (w_c) respectively.

$$wsr(w_t, w_c) = \max_{c_t, c_c}[srm(c_t, c_c)] \tag{3.6}$$

Each score was then normalized to fall on a scale of one to ten. This was done by first determining the maximum possibility similarity score for each profession category. The ratio of the similarity score of the word to the maximum possible profession similarity score was then multiplied by ten. The maximum possible word similarity score for a profession category was calculated as the word similarity between the category and itself. For example, the word similarity score between "athlete" and "athlete" was 6.789, making it the maximum possible score any term could score against "athlete". The equation for the normalized score is defined in Equation 3.7.

$$wsr_{norm}(w_t, w_c) = \frac{wsr(w_t, w_c)}{wsr(w_c, w_c)} * 10 \tag{3.7}$$

The final score was then calculated as the sum of all the normalized word similarity scores in the original tweet for each potential profession, as defined in the person dictionary. The profession category with the highest resulting score from the disambiguation was then used as the

classification for the person entity in question. The equation used for this calculation is defined in Equation 3.8.

$$sim\_score_c = \sum_i wsr_{norm}(w_i, w_c) \quad (3.8)$$

Using Freebase to Classify an Entity

Because of the relative success at using the web to classify an entity and to create a person dictionary, using an already existing gazetteer such as freebase (Bollacker, Evans and Paritosh) seemed to be a good next step. Freebase is a large, online database that is populated and maintained by its community of online users. In this method of classifying the entity, a call to freebase was made using the freebase API that filtered on people data. Freebase returned the top hits on the query and then the best database entry was selected from those returned. Then from that entry, a profession category was selected and used to classify the entity. Several methods of selecting the best entry and selecting a profession category were investigated and are described within this section.

Selecting the Best Freebase Entry

When a query is made through the freebase API, freebase returns the best N results to the user where N is a parameter in the search that defines how many entries to return. For this system, N was set to ten. Several options for selecting the best entry out of those returned entries were investigated.

The first option was to select the entry recommended by freebase. Each freebase entry contained a score that rated the particular entry on how relative it was to the query, as ranked by the freebase user community. So in this option the system simply selected the entry with the highest score from freebase. While this seemed a good option, there were many instances where the most relative entry as determined by freebase was not the correct entry for an entity, so other methods of selecting the best entry were explored.

The second option was to do a word match count on the nouns in the tweet and the nouns in the freebase tags. The freebase return objects contain a list of tags that describe the entity. A match count to see how many of these tags matched up with nouns in the tweet was done and the entry that had the most matches was selected. In a case where no entry had a match, then the freebase recommended entry was the one selected as the best match. In the cases where there was a tie, then the freebase recommendation score was used as the tiebreaker.

The third option was to do a string match count instead of a word match count. This was similar to the word match count option, but differed in that it counted string matches instead of words. In order to qualify for a string match, the first two letters must match between one of the freebase tags and one of the nouns from the tweets. Then the string match was calculated as the number of consecutive matching letters divided by the number of letters in the tweet noun. The cumulative score for all tags against all tweet nouns was calculated and the entry with the highest score was selected as the best entry. Once again, the freebase recommendation score was used as the tiebreaker. Additionally, if there were no entries that got a string match score, then the freebase recommended entry was used as the default best entry.

The last option was to do a word match count that also included the glosses derived from WordNet (Miller). In this option, a word match count is done using the tags from the freebase entries and the nouns from the tweet. But in addition to that, a word match count was used on the glosses of each tag against the glosses of each tweet noun as well. The glosses were derived by using WordNet to get the synonyms for each tag and each noun. This was tested using only the first level glosses (the synonyms for each tag and each noun) as well as using the second level glosses (the synonyms for each word in the first level glosses). The entry that had the best cumulative score was then selected as the best entry. If no entry was selected, then the default entry was the freebase recommended entry. If multiple entries had the same word match count, then the freebase recommendation score was used as the tiebreaker.

Selecting the Best Profession

Once a freebase entry was selected as the best match entry for the entity to be classified, the next step was to derive the profession category from data contained within the freebase entry. There were three options in the data that could be used to derive the profession. The first was the notable field, a field in the freebase object that defined what the person was most notable for. There was also the set of type tags which were used in some of the entry selection methods. These tags contained a list of all potential classifications for the entity. The last possibility was to use the description text. This text was usually a brief biographical blurb that was either written by a freebase user or taken from a Wikipedia article of the same entity. This entry was sometimes from foreign Wikipedia sites and was not always useful as a result.

The first method investigated was to use only the notable field. The notable field was looked up in the same IS-A relation file that was used in the web query method to determine which profession category the notable field corresponded to and then the classification was made based on that. The notable field in freebase is assigned by users in the freebase community and is meant to be what the person is most notable for. However, since the notable field did not always correspond to one of the profession categories, or sometimes said the person was notable for something other than what would have been considered correct by our system, other methods of extracting a profession category from the entry were explored. One example of such a case was the entry for Mitt Romney which had him as notable for being a businessman while the corpus was gathered during Mitt Romney's presidential run, so the system considered a classification of politician as the correct one.

The second method was to use the type tags to determine possible person classifications. In addition to using the type tags, the notable field was also used as a tag in this method. Each tag was looked up in the IS-A relation file to see if it matched a profession category. A count was kept that counted each occurrence of each profession category, and the profession category that had the most entries from among the tags was then used to classify the entity. In an alternative of this method, each potential category was saved, and then the disambiguation method from the dictionary method was used to determine the best profession category for classification, but this approach was discarded as the added ambiguity was counterproductive.

The last method was to use the description text to determine the best profession. In this method, the same pattern matching that was used to parse the web results was used on the freebase

description. All potential profession categories were extracted, and then the disambiguation algorithm from the person dictionary method was used to select the best profession.

Hybrid Methods

Each of the three main methods used for classification were found to be moderately successful at correctly assigning a profession classification, but each method of classification had strengths that seemed complimentary. The classifier was good at assigning a profession category when there was enough content in the tweet. The person dictionary method was good at assigning a classification if there was no ambiguity, as was the freebase method. This led to an investigation of several hybrid methods that combined various pieces of each of the method in an attempt to better classify the entity into one of the profession categories.

Dictionary-Classifier Hybrid

In this hybrid method, the dictionary was used as the first classification option if an entity had only one entry in the dictionary. If the dictionary had multiple entries for the person, then the classifier's classification was used provided it had a classification confidence level greater than a configurable threshold, which was experimentally found to be 0.6. If the confidence level was less than the threshold, then the disambiguation method was used on the potential profession categories defined in the dictionary for the entity to determine the classification. If the dictionary did not contain an entry for the entity and the web search also did not return a profession, then the classifier's classification was used regardless of its confidence level.

The basis of this hybrid method was the idea that the strength of the dictionary lies in its ability to search the web and find information for an entity regardless of how much content there was in the tweet. If the web search came back and a search through all the web results only yielded one potential profession category for a person, then it seemed most likely that that profession category derived from the search was the correct one. In situations where the dictionary or web search results returned more than one profession category option, then the disambiguation method of the dictionary would need to be run. But the disambiguation method required content in the tweet in order to work effectively and any content that would help with the disambiguation was also the content that the classifier would use to make a classification. In this situation, the classifier was generally more accurate than the disambiguation method when it had a reasonable level of confidence in the classification. So this hybrid method was an attempt to use the strengths of both methods to improve the overall level of accuracy of the profession classification.

Freebase-Classifier Hybrid

This hybrid method was basically the same as the dictionary-classifier hybrid, only it used the freebase call in place of the dictionary lookup/web query. The strengths of freebase compared to the classifier were basically the same, with the strength being that when freebase only has one classification to choose from, it does a better job than the classifier. But the classifier was better in situations where freebase had to decide between multiple profession categories, so this hybrid attempted to use the strengths of both methods, just as in the dictionary-classifier hybrid method.

Dictionary-Classifier-Freebase Hybrid

This hybrid was similar to the first two, with one key difference. In this hybrid method, the dictionary method was used if an entity had only one potential profession category returned from the dictionary (or web search results). Otherwise, it used the classifier's classification so long as the classifier's confidence level was above the configurable threshold. The difference of this method was what happened when the classifier did not have a high level of confidence in its classification. Rather than use the disambiguation method on the categories from the dictionary, a call was first made to freebase and the freebase method was used to make the classification. If the freebase method didn't return a profession, then the disambiguation algorithm was run on the categories from the dictionary. In cases where the dictionary didn't have an entry for an entity and the web query did not return any potential profession categories, then freebase was used. In the event that freebase also did not return a hit, then the classifier's classification was used regardless of the confidence level of the classification.

During the testing of the various methods, it was discovered that in some cases, the freebase method did a better job of classifying entities when there was ambiguity than the dictionary method did. That is what this hybrid method was designed to try to take advantage of by first using the freebase call in cases where otherwise the dictionary disambiguation would have run.

Freebase-Classifier-Dictionary Hybrid

This method is the same as the Dictionary-Classifier-Freebase method, only in this case the freebase method was used first. If freebase did not return a profession assignment, then the

classifier was used if its confidence level was more than the required threshold. Otherwise, the dictionary method was used to make the classification.

Dictionary-Freebase-Classifier Hybrid

In this hybrid method, a look up to the dictionary was performed first. If the dictionary method only returned one profession, then that classification was used for the entity. If there was ambiguity in the dictionary, then a freebase call was made. If freebase returned a profession, that was used. If freebase did not return a profession category, then the classifier was used if the confidence in the classification was greater than the configured threshold. If it was not, then the disambiguation method was run on the dictionary entries, so long as the entity had dictionary entries. If the entity wasn't in the dictionary and a web search also didn't produce a profession classification, then once it got to this point, the classifier's classification was used regardless of the confidence level.

Freebase-Dictionary-Hybrid

This method is the same as the Dictionary-Freebase-Hybrid method, with the difference being that the freebase call was performed first, and the dictionary method was called only if freebase didn't return a profession category. If the dictionary contained only one profession category, then that was used as the classification, but if the dictionary contained ambiguity or if the dictionary method did not return a profession, then the classifier was used if its confidence level was larger than the threshold. If the confidence level was too low, then the disambiguation method was run on the potential profession categories from the dictionary if the dictionary method had produced

any. If the dictionary method did not have any profession options, then the classifier's classification was used regardless of the confidence level.

CHAPTER FOUR: RESULTS

To test the classifier, the classifier was trained on several different combinations of features and then tested using the validation and hold out sets described in the data set generation section of this paper. The classifier was also run with and without the high confidence classification table described in the classifier section of the methodology chapter.

Table 5
A Look at the Accuracy of the Classifier using Different Feature Sets

Classifier Features	Validation Set	Hold Out Set
WF, TSS	62.1%	50.2%
WF TSS, HCCT	71.7%	56.3%
WF, TSS, HCCT, GP	78.2%	67.1%
WF TSS, HCCT, GP, HW	80.1%	71.1%

WF = Word Frequency Features.
TSS = Topic Signature Scores Features
HCCT = High Confidence Classification Table
GP = Generalized Patterns and Pattern Weights
HW = Hashtag Weights

The results show the effects of the different combinations of features. Most notably, the assumption that entities trend in twitter, which allowed for the use of the high confidence classification table, helped substantially. The generalized patterns and weights also had a large effect on the accuracy of the classifier. The next table looks specifically at how the accuracy of classifications into each profession category was affected by the generalized patterns to determine why the improvement was so dramatic.

Table 6
A Look at the Effect of Using the Generalized Patterns

Profession	Validation Set Before	Validation Set After	Hold Out Set Before	Hold Out Set After
ATHLETE	82.3%	82.0%	77.0%	77.7%
AUTHOR	50.0%	62.3%	65.3%	77.3%
BUSINESS	0.0%	22.2%	12.5%	77.1%
CLERGY	81.3%	84.8%	59.3%	69.5%
DOCTOR	0.0%	20.0%	11.1%	59.3%
ENTERTAINER	70.6%	78.8%	52.7%	56.2%
JOURNALIST	50.0%	66.7%	51.1%	78.4%
POLITICIAN	60.0%	72.0%	59.8%	78.9%

Using the generalized patterns improved the performance of the classifier in almost every instance for each individual profession category in each set. Within the training data, there were quite a few book reviews, so the <RATING> tag appears to have been effective in improving the Author classifications. Additionally, the <REFERENCE> tag seems to have played a role in the improvement of the Clergy classifications as well. The most surprising results are the relatively small impact on the Athlete classifications as well as the improvement across many other categories. Further breaking down the results, it turns out that the primary driver of the improvements in the other categories was a substantial decrease in the number of incorrectly classified Athletes in the final set. In other words, prior to the addition of the generalized patterns to the classifier, the classifier incorrectly classified far more entities as athletes than after the patterns were included. This resulted in more entities being properly classified as the <SCORE> tag became an indicator of the Athlete profession and the absence of a <SCORE> tag led to a decrease in the number of incorrectly classified Athletes. A summary of the decrease in incorrectly classified Athletes is summarized in the next table.

Table 7
A Look at the Effect of Using Generalized Patterns on False Athlete Classifications

	Percentage of False Athlete Classifications in Incorrect Classifications in Validation Set	Percentage of False Athlete Classifications in Incorrect Classifications in Hold Out Set
Without Generalized Patterns	59.4%	61.4%
With Generalized Patterns	53.6%	44.3%

The next table breaks down the effectiveness of the dictionary classification method. It compares the results of the classifier using the dictionary that was manually compiled for use in the bootstrapping testing data generation method against the dictionary that was automatically generated though applying the pattern matching on the web search results.

Table 8
A Look at the Accuracy of the Person Dictionary Method

	Validation Set	Hold Out Set
Automatically Generated Dictionary	86.4%	78.4%
Manually Compiled Dictionary	90.7%	85.7%

In addition to being more extensive, the manually compiled dictionary had the advantage of having human insight as to whether or not an entity should have a profession category added to the dictionary file or not. This led to less ambiguity in the manually compiled dictionary despite its larger size. Given the advantages of the manually compiled dictionary, the results indicate that the automatically generated dictionary performed reasonably well in comparison. Ambiguity was the biggest factor in decreasing the accuracy in the dictionary method and the next table provides

a data on how the accuracy of the method was affected as more profession categories were available for an entity. The more ambiguous the person, the less accurate the dictionary became.

Table 9
A Look at How Ambiguity Affects the Person Dictionary Classification Method Accuracy

Number of Profession Category Entries	Validation Set	Hold Out Set
1	96.7%	97.6%
2	77.3%	72.3%
3	72.2%	55.6%

The next table shows the accuracy of the various freebase methods described in this paper. It is broken down by the method used to select which freebase entry to select from the results and which method used to extract the profession category from that freebase entry. In general, no combination of entry selection criteria and profession selection criteria performed better than just using the freebase suggested entry and profession, hence in the hybrid tests, the entry was selected according to the freebase relevance score and the entry was chosen as the freebase notable field. This is not to be unexpected as the freebase relevance score and the notable fields are based on user rankings. Comparing freebase against the dictionary, using the freebase suggestions outperformed the automated dictionary on the validation set but did not do as well on the hold out set and in general, the freebase methods did not do as well as the dictionary method, especially with the hold out set.

Table 10
A Look at the Accuracy of the Various Freebase Methods

Freebase Entry Selection	Profession Selection	Validation Set	Hold Out Set
Freebase Best	Notable	87.4%	74.9%
Freebase Best	Type Count	86.6%	74.4%
Freebase Best	Pattern Matching	85.4%	70.0%
Word Match Count	Notable	87.2%	74.9%
Word Match Count	Type Count	88.0%	75.8%
Word Match Count	Pattern Matching	84.9%	69.9%
String Match Count	Notable	80.6%	63.5%
String Match Count	Type Count	80.9%	62.5%
String Match Count	Pattern Matching	79.0%	59.8%
One Level Gloss	Notable	86.4%	74.8%
One Level Gloss	Type Count	75.4%	62.3%
One Level Gloss	Pattern Matching	85.1%	69.1%
Two Level Gloss	Notable	74.6%	60.4%
Two Level Gloss	Type Count	77.0%	63.1%
Two Level Gloss	Pattern Matching	85.0%	69.0%

The next table looks at the results of the various hybrid methods in comparison to the original methods and shows that using freebase alone in the hybrid is not as good as using the dictionary, which suggests that the dictionary is probably a slightly more accurate method. The results also show that the inclusion of freebase in any of the classifiers tended to decrease the accuracy of the classifications in the hold out set. This is probably because the validation set came from the bootstrapping method, which relied upon internet lists and high confidence classifications to produce the data. As a result of that, we may conclude that the validation set most likely contains more famous and more easily identifiable people than the hold out set. These famous people are more likely to have been correctly categorized in freebase and to have relevancy scores that are more accurate in freebase since it is content that is probably viewed more often by the freebase users.

Table 11
A Comparison of the Accuracy of All Classification Methods

Classification Method	Validation Set	Hold Out Set
Classifier	80.1%	71.1%
Automated Dictionary	86.4%	78.4%
Freebase	88.0%	76.0%
Dictionary-Classifier Hybrid	92.3%	88.0%
Freebase-Classifier Hybrid	89.9%	82.1%
Dictionary-Classifier-Freebase Hybrid	92.9%	87.8%
Freebase-Classifier-Dictionary Hybrid	90.2%	82.6%
Dictionary-Freebase-Classifier Hybrid	93.1%	86.4%
Freebase-Dictionary-Classifier Hybrid	85.6%	83.4%

The last thing investigated was how time would degrade the results of the classifier. One of the potential strengths of the person dictionary method is that if a person is trending on Twitter, that person is also probably trending on the internet. Thus, any web query on that person would likely return several web search results that contain not only the correct instance of the person, but also the correct context for that person. This would reasonably result in an increase in the likelihood of extracting the proper profession category as well as a decrease in the likelihood of finding ambiguous entries. But the more that time passes between the collection of the tweet and the generation of the dictionary, the less likely the web results will take advantage of that simultaneous trending quality, which would result in less accurate classifier results. In order to test this hypothesis, a new person dictionary was automatically generated in May of 2013, a little more than a year after the collection of the original tweet corpus. The following table shows the performance of the automated dictionary algorithm, as well as the classifier-dictionary hybrid, when the newer dictionary was used in place of the original one that was generated much closer

to the original tweet corpus collection. The results show that the accuracy of both methods decreased when the new dictionaries were used.

Table 12
A Look at how Time Degrades the Performance of the Dictionary Method

	Validation Set	Hold Out Set
Original Automatically Generated Dictionary	86.4%	78.4%
New Automatically Generated Dictionary	85.9%	76.7%
Classifier-Original Dictionary Hybrid	92.3%	88.0%
Classifier-New Dictionary Hybrid	90.7%	82.6%

CHAPTER FIVE: CONCLUSIONS

Each of the classification techniques has their individual strengths and weaknesses, and the hybrid methods tend to perform better because they are able to combine those strengths. For the classifier, its strengths are in classifying tweets that are longer, and thus have more useful content than the shorter tweets. An additional strength of the classifier is that it can classify an entity correctly regardless of whether or not that person is famous if the tweet contains enough information. The classifier is also helped by twitter specific features such as hash tags. The classifier's weakness is that regardless of the training set, it is going to struggle to classify shorter tweets. And since tweets are already limited in length to begin with, this is especially problematic.

The automated dictionary method's strength is that it can accurately classify entities regardless of the length of the tweet because it uses the web as a means to make the classification. Another advantage of the web search is that it is quick to update, and the web results trend right along with Twitter. This means that people who are showing up in Twitter are probably showing up in news stories across the web as well, and the web results that come back from the query are likely to contain hits on the correct person in the correct context to make an accurate classification. In order to take advantage of this particular strength in a live system, one possible modification to the algorithm would be to have the dictionary entries expire after a certain amount of time or to weight the entries in the dictionary by time. The weaknesses of the dictionary method are that its accuracy goes down when there is ambiguity in the dictionary or the web search results, and that if a first name and a family name are not contained in the tweet, the web search results will most

likely not return useful data unless it is somebody famous enough to be widely known by one name like Madonna. Also, if a tweet is about an average person and not somebody famous, the webs search will probably not return a match and then the dictionary won't have an entry for that person. One additional weakness is that if you have a static testing set, the dictionary is less accurate as more time has elapsed between the set compilation and the dictionary compilation. For this reason, creating a dictionary file to be used as a reference online might prove to be less effective than creating and maintaining a dictionary automatically through use of the algorithm as names come up.

Freebase has a similar strength as the dictionary method in that is contains a wealth of information and is a good way to get additional context on a person than is available in the tweet. It also is very large and contains more variety in the return set than a web query result might have. This is both a potential strength and weakness though when compared to the dictionary method. On the one hand, a really obscure person might be more likely to come up in the freebase method, which is good. But on the other hand, the added variety in the result set ends up causing more ambiguity, and is the most likely cause of the freebase method not performing as well as the dictionary method. Another weakness of the freebase method is that is does not update as quickly as a web search does. For instance, the Boston bomber suspect Dzhokhar Tsarnaev is still tagged as only a person in the freebase database (as of Jun 17, 2013), whereas a web search conducted on him in the days after the bombing and even now contains results from which it can be determined that he is a bombing suspect.

The results of the various hybrid methods show that by maximizing the strengths of each method, the accuracy of the classification can be improved substantially. For example, the dictionary-classifier method is able to maximize the strength of the automated dictionary method by using its classification when the results all agree on the classification. Additionally, this makes up for the weakness of the classifier in the sense that it doesn't matter how big the tweet is when this method is used. This hybrid also takes advantage of the classifier's ability to classify entities without a full name and people who are not famous, provided sufficient context is provided, which makes up for that weakness of the dictionary. And when there is ambiguity from the web search or dictionary, the classifier does a better job of classifying that entity if there is sufficient context for the classifier to be confident, which also minimizes the weakness of the dictionary method.

The results show that the hash tag weighing and the generalized pattern features and weights improved the quality of the classifier, as did the use of the high confidence classification table. The results also show that the automated dictionary method compares favorably to the use of a manually compiled dictionary. Lastly, while each individual method performed reasonably well, the hybrid methods were able to take advantage of the strengths of the various methods and improve the accuracy of the system considerably. The results indicate this is a promising method of classifying entities by profession.

APPENDIX A:
PROCESSING THE TWEET DATA

Twitter data is returned in a JSON object format. The purpose of this appendix is to illustrate how the system processes the raw twitter stream data to prepare it for use in the classification system. An example of the raw data is seen below. Please note that the user information and id information has been removed as a precaution.

{"geo":null,"in_reply_to_status_id":null,"text":"Blake Griffin and I have the same Final Four and overall winner. Ment 2 be? #MarchMadness","in_reply_to_user_id":null,"in_reply_to_status_id_str":null,"retweet_count":0,"truncated":false,"id_str":"xxxxxxxxxxx","entities":{"urls":[],"hashtags":[{"text":"MarchMadness","indices":[77,90]}],"user_mentions":[]},"contributors":null,"in_reply_to_user_id_str":null,"place":null,"retweeted":false,"source":"\u003Ca href=\"http:\/\/twitter.com\/download\/android\" rel=\"nofollow\"\u003ETwitter for Android\u003C\/a\u003E","coordinates":null,"in_reply_to_screen_name":null,"user":{},"favorited":false,"id": xxxxxxxxxxx,"created_at":"Wed Mar 14 22:09:38 +0000 2012"}

The first step is to extract the user id, tweet id, hashtags, and text of the tweet. The user id and tweet id are used to identify the tweet. The hashtags are saved in order to be used in the hash tag weighting part of the algorithm and the text is saved to be run through the text processing. The text of the tweet is “Blake Griffin and I have the same Final Four and overall winner. Ment 2 be?”

The first step is to preprocess the text using the abbreviations, contractions, and misspellings dictionaries that were compiled. Using these resources, the tweet text is cleaned up to read, “Blake Griffin and I have the same Final Four and overall winner. Meant to be?”
Next, the tweet is stemmed, tokenized, and entity tagged. Within the system, the stemmed and tokenized version of the text is used for the word frequency feature calculations; the preprocessed tweet text is saved for human readability and for use in searches. The processed tweet then reads, “<PERSON>blak griffin</PERSON> and i have the same final four and overal

winner . meant to be ? # marchmad" and the human readable copy of the tweet reads, "<PERSON>Blake Griffin</PERSON> and I have the same Final Four and overall winner . Meant to be ? # MarchMadness" At this point the tweet is ready to be assigned a profession category by the system.

APPENDIX B:
PROCESSING WEB QUERY DATA

Given the input “<PERSON>Blake Griffin</PERSON> and I have the same Final Four and overall winner . Meant to be ? # MarchMadness”, the automated dictionary method extracts the person “Blake Griffin” and prepares a web query data object to be used by the Yahoo BOSS! search API. An excerpt of what the web results return is seen below.

> {"results":[{"abstract":"Blake Austin Griffin (born March 16, 1989) is an American professional basketball player who currently plays for the Los Angeles Clippers of the National Basketball ...","clickurl":"http://en.wikipedia.org/wiki/Blake_Griffin","title":"Blake Griffin<\/b> - Wikipedia, the free encyclopedia","dispurl":"en.wikipedia.org/wiki/Blake_Griffin<\/b>","date":"","url":"http://en.wikipedia.org/wiki/Blake_Griffin"},{"abstract":"Blake Griffin<\/b>, LA Clippers, NBA, Oklahoma ... Blake slammed 214 dunks this season - that's over $21,400 going to combat childhood obesity!","clickurl":"http://www.blakegriffin.com/pages/main","title":"Blake Griffin<\/b> | The Official Web Site","dispurl":"www.blakegriffin<\/b>.com/pages/main","date":"","url":"http://www.blakegriffin.com/pages/main"}}

For each of the results, the title of the page and the description are parsed out and then checked for pattern matches. For example, the title of the first result in the above entry is “Blake Griffin - Wikipedia, the free encyclopedia” and the description is “Blake Austin Griffin (born March 16, 1989) is an American professional basketball player who currently plays for the Los Angeles Clippers of the National Basketball ...” The title and the description are both then checked for matches and one is found in the description for the text, “Blake Austin Griffin (born March 18, 1989) is an American professional basketball player.” The pattern match extracts basketball player as a potential profession, and then uses the IS-A file to look up basketball player to see if it matches a profession category. Basketball player returns a match for the Athlete category, so an entry is made in the person dictionary for Blake Griffin as an Athlete. This process is repeated for every web result and if at the end of it, the only entry in the

dictionary for Blake Griffin is Athlete, then Blake Griffin is labeled as an Athlete in the preprocessed tweet like so,"<ATHLETE>Blake Griffin</ATHLETE> and I have the same Final Four and overall winner . Meant to be ? # MarchMadness".

APPENDIX C:
PROCESSING FREEBASE DATA

The method of pulling a result out of freebase is similar to the method of performing a web query. Given a tagged tweet as input such as “<PERSON>Blake Griffin</PERSON> and I have the same Final Four and overall winner . Meant to be ? # MarchMadness”, a freebase query for the person Blake Griffin is constructed. An example of the first three results returned can be seen below.


```
{  "status": "200 OK", "result": [
{"mid": "/m/02qhzxp", "id": "/en/blake_griffin","name": "Blake Griffin",
"notable": {
        "name": "Basketball player",
        "id": "/m/02h664x"
},"lang": "en", "score": 595.830994,
"output": {
        "description": {
                "wikipedia": [
                        "Blake Austin Griffin is an American professional basketball
                        player for the Los Angeles Clippers of the National Basketball
                        Association. Griffin had a renowned high school career at
                        Oklahoma Christian School, winning state titles each of his four
                        years under his father, Tommy Griffin, who was the head coach.
                        He played college basketball for the University of Oklahoma
                        Sooners and received numerous accolades in his second year,
                        including the Naismith College Player of the Year, Oscar
                        Robertson Trophy and the John Wooden Award.\nGriffin left
                        college after two seasons to enter the 2009 NBA Draft; he was
                        selected first overall by the Clippers. In his first season, he broke
                        his left kneecap during the final pre-season game, had surgery, and
                        missed the entire 2009\u201310 season. He made his NBA debut
                        as a rookie the following season, in which he was selected as an
                        All-Star, won the 2011 NBA Slam Dunk Contest, and was named
                        the NBA Rookie of the Year. In 2011, Sports Illustrated called him
                        one of the NBA's 15 Greatest Rookies of All Time." ]}}},
{"mid": "/m/0hb8cvj","id": "/authority/imdb/name/nm2968223", "name":"Blake Griffin",
"notable": {
        "name": "Actor",
        "id": "/m/02hrh1q"},
"lang": "en", "score": 165.632874,
"output": {
        "description": {
                "freebase": [
                        "Blake Griffin is an actor."]} }},
```

{"mid": "/m/052lgq", "id": "/en/w_e_b_griffin","name": "W. E. B. Griffin",
"notable": {
"name": "Writer",
"id": "/m/0cbd2"},
"lang": "en",
"score": 29.054258,
"output": {
"description": {
"wikipedia": [
"W. E. B. Griffin is a writer of military and detective fiction with 38 novels in six series published under that name. He has also published under several pseudonyms."] }}}
],
"cursor": 10,
"cost": 6,
"hits": 45
}

The fields that are important to the algorithms are the name field, the notable field (in which the name is parsed out), the score field, and the output field under which the wikipedia or freebase description can be obtained. These examples do not contain any of the type tags that were used in some of the freebase methods. From the results, the first “Blake Griffin” scores the best score and is what would have been used by default. The notable field identifies that Blake Griffin is a basketball player, which would then be checked for a matching profession category in the IS-A reference file. The lookup would then return that a basketball player is an athlete, and Blake Griffin would be classified as an athlete in the original preprocessed tweet as follows, “<ATHLETE>Blake Griffin</ATHLETE> and I have the same Final Four and overall winner . Meant to be ? # MarchMadness”.

REFERENCES

Alex, B., A. Dubey and F. Keller. "Using Foreign Inclusion Detection to Improve Parsing Performance." *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. 2007.

Apache Open NLP. n.d. 12 March 2012. <http://opennlp.apache.org>.

Bolken, J., H. Mao and X-J Zeng. "Twitter mood predicts the stock market." *Proceedings of the Fourth International Conference on Weblogs and Social Media (ICWSM-2010).* Washington, DC, 2010.

Bollacker, K., et al. "Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge." *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*. Vancouver, Canada, 2008. 1247-1250.

Cucerzan, S. "Large-Scale Named Entity Disambiguation Based on Wikipedia Data." Prague, Czech Republic, 2007. 708-716.

Dunning, T.E. "Accurate methods for statistics of surprise and coincidence." *Computational Linguistics* (1993).

Efron, M. "Information Search and Retrieval in Microblogs." *Journal of the American Society for Information Scient and Technology* (2011).

Finin, T., et al. "Annotating named entities in Twitter data with crowd sourcing." *Proceedings of the NAACL Workshop on Creating Speech and Text Language Data with Amazon's Mechanical Turk.* 2010.

Fleischman, M. and E. Hovy. "Fine Grained Classification of Named Entities." *Proceedings of COLING-2*. 2002.

Fleischman, M. "Automated Subcategorization of Named Entities." *Proceedings of the ACL Student Workshop*. 2001.

Gimpel, K., et al. "Part-of-Speech Tagging for Twitter: Annotation, Features, and Experiements." *Proceedings of ACL 2011*. 2011.

Gouws, S., et al. "Contextual bearing on linguistic variation in social media." *ACL Workshop on Language in Social Media*. Portland, OR, 2011.

Hall, A. and F. Gomez. "Using an automatically generated Dictionary and a Classifier to Identify a Person's Profession in Tweets." *Proceedings of The 26th International Florida Artificial Intelligence Research Society Conference*. St. Pete Beach, FL, 2013.

Han, B. and T. Baldwin. "Lexical normalisation of short text messages: Makn sens a #twitter." *The 49th Annual Meeting of the Association for Computational Linguistics*. Portland, OR, 2011.

Hearst, M. "Automatic Acquisition of Hyponyms from Large Text Corpora." *Proceedings of the Fourteenth International Conference on Computation Linguistics*. 1992.

Higashinaka, R., et al. "Creating an Extedned Named Entity Dictionary from Wikipedia." *Proceedings of COLING-12: Technical Papers*. Mumbai, India, 2012. 1163-1178.

Jackoway, A., H. Samet and J. Sankaranarayanan. "Identification of Live News Events using Twitter." *Proceedings of ACM SIGSPATIAL International Workshop on Location-Based Social Networks*. Chicago, IL, 2011. 25-32.

Li, C., et al. "TwiNER: Named Entity Recognition in Targeted Twitter Stream." *Proceedings of SIGIR 2012*. Portland, OR, n.d.

Lin, C-Y and E. Hovy. "The Automated Acquisition of Topic Signatures for Text Summarization." *Proceedings of COLING-00*. 2000.

Liu, X, S Zhang and F Wei. "Recognizing Named Entities in Tweets." *Proceedings of the Association for Computational Linguistics: Human Language Technologies*. 2011.

Locke, B. and J. Martin. "Named Entity Recognition: Adapting to Microblogging." Master's Thesis. 2009.

Maynard, D. and A. Funk. "Automatic detection of political opinions in Tweets." *The Semantic Web: ESWC 2011 Selected Workshop Papers, Lecture Notes in Computer Science*. 2011.

Miller, G. "WordNet: a lexical database for English." *Commmunications of the ACM* (1995): 39-41.

Nadeau, D. and S. Sekine. "A Survey of Named Entity Recognition and Classification." *Lingvisticae Investigationes* (2007).

Nadeau, D. "Semi-Supervised Named Entity Recognition: Learning to Recognize 100 Entity Types with Little Supervision." Ottawa, Canada: Ottawa-Carleton Institude for Comp;uter Science School of Information Technology and Engineering, University of Ottawa, 2007.

Nothman, J. "Learning Named Entity Recognition from Wikipedia." Sydney, Australia: School of Information Technologies, The University of Sydney, 2008.

Owoputi, O., et al. "Part-of-Speech Tagging for Twitter: Word Clusters and Other Advances." Technical Report CMU-ML-12-107. 2012.

Porter, M. "An algorithm for suffix stripping." *New models in probabilistic information retrieval* (1980).

Quinlan, J.R. *C4.5: Programs for Machine Learning*. 1993.

Resnik, P. "Selection and Information." PhD Thesis. 1993.

Ritter, A., S. Clark and O. Etzioni. "Named Entity Recognition in Tweets: An Experimental Study." *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. 2011.

Schwartz, H., F. Gomez and L. Ungar. "Improving Supervised Sense Disambiguation." *Proceedings of COLING-12*. Mumbai, India, 2012.

The Porter Stemming Algorithm. n.d. 29 March 2012. <http://tartarus.org/~martin/PorterStemmer/>.

The Stanford Natural Language Processing Group. n.d. 30 March 2012. <http://nlp.stanford.edu/software/CRF-NER.shtml>.

Toutanava, K., et al. "Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency." *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Edmonton, Canada, 2003. 252-259.

Twitter Blog. n.d. 29 March 2012. <http://blog.twitter.com/2011/03/numbers.html>.

Twitter4J. n.d. 1 April 2012. <http://twitter4j.org>.

UnBBayes - The UnBBayes Site. n.d. 3 April 2012. <http://unbbayes.sourceforge.net>.

Yahoo! BOSS Search. n.d. August 2012. <http://developer.yahoo.com/boss/search>.

Yahya, M., et al. "Natural Language Questions for the Web of Data." *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. 2012. 379-390.