ABSTRACT

Twitter is a highly accessible, diverse and representative body of user-generated content reflecting public opinion and sentiment in near-real time. This makes it an ideal source of unstructured data for generating predictions in advance of outcomes for certain kinds of events. Large-scale analysis of unstructured, human-generated data can offer online insight into some remarkably specific, domain-constrained predictions (e.g., theater box office sales, restaurant choices, travel destinations, etc.).

This system addresses the question of whether Twitter sentiment can be used in a predictive capacity to determine the relative ranking of elements within a topic domain and applies it to one in particular (the New York Times fiction bestseller list). Fundamentally, the system designed relies on sentiment and frequency analysis as a means to score the relative weighting of elements of a topic domain.

1. INTRODUCTION

The objective is to use Twitter to analyze topic domains that generate nontrivial mention frequency in social media and extrapolate that analysis into relational predictions. Categorical information domains (e.g., movies, TV shows, foods, etc.) are ripe for the kind of relational analysis this system is capable of conducting. Social media outlets that are publicly accessible can be used to survey human-generated contextual information related to such domains. Furthermore, this unstructured data can be assessed according to a variety of statistical and linguistic domains. This system relies primarily on two of these: raw frequency of specific instances and sentiment classification. These features alone can be used to draw powerful conclusions about relative sentiment with respect to various topic domains.

For this purpose, a specific application of the aforementioned relational predictive analysis had to be chosen. The benefit of focusing exhaustively on one application meant that the system designed could later be abstracted to other domains once calibrated on a given domain. The primary criteria for choosing a domain for analysis included: relatively high mention frequency and access to data for labeling outcomes. Secondary criteria included assessing a domain that is not normally approached by researchers and that would also yield results fruitful for some potential third party. The specific domain chosen was to determine week-by-week fluctuations in the relative favorabilities of fiction novel bestsellers. In the realm of literature, sales data acts as a sufficient proxy for favorability and the New York Times Fiction Bestseller List provides exactly this information. The system was therefore applied towards predicting the week-by-week movements of novels on the New York Times list. Our motivations in choosing this domain are further expounded upon in Section 2.

The outcome of approaching this problem domain was a modular software of sufficient complexity to meet all the demands involved in producing this relational analysis. The core components of the system were the Collector, Filter, and Ranker. The Collector plugged into the Twitter ‘firehouse’ to continuously draw, compress, and store a sampling of all tweets generated on Twitter in the period for which the Collector was active. The Filter operated on the massive dataset mined by the Collector and acted to reduce it via contextual analysis to exactly the set of Tweets relevant to the topic domain. Finally, the Ranker used both statistical and sentiment analysis to analyze the filtered set and generate the necessary relational predictions. Crucially, the system designed is structured in a manner that is sufficiently scalable across large data sets (in this case Twitter data), abstractable to other problem domains and extensible in a way that allows easy modification or refinement of components of the analysis. Furthermore, the application of the system to predicting relative favorability of literature acted as a calibration mechanic so that it might be applied to other domains that are unlabeled, given the correct inputs.

2. BACKGROUND

The explosion of publicly available, anonymized and unstructured data generated by human participants through Twitter, online blogging, and other digital mediums of discourse has made performing Natural Language Processing (NLP) and sentiment analysis efficiently at scale an increasingly more viable task. As stated, the initial core objective of the system designed was to utilize Twitter sentiment analysis to predict the composition of the New York Times Best Sellers List of fictional works on a week-by-week basis. The New York Times Best Sellers List is the preeminent best seller list in the country. It is divided into several categories and is released weekly on Sundays and earlier online [23]. The motivation for this system is predicated on the fact that a leading indicator of sales rankings (e.g., theater box
3. RELATED WORK

Twitter has been very popular as a web service and since its inception many researches have been focused on analyzing tweet sentiments to predict events such as stock market movements [5] and box office receipts [2]. Recent research indicates that interest and engagement in popular events is typically tracked by fluctuations in public sentiment, as seems intuitively likely [11]. Furthermore, these results motivated subsequent studies to relate social media data to real world events. Consequently, various approaches have been proposed to conduct sentiment-based Internet time series analysis, as explained in the following subsections.

3.1 Sentiment Analysis of Online Text and Time Series Analysis of Online Topics

With the prevalence of web services, more and more data are available to researchers to automatically measure emotion in online texts. The Natural Language Processing (NLP) community has developed systems to automate Sentiment Analysis [14, 16]. Freely available tools, such as NLTK [4] which is one of the most widely used NLP packages, are used by many academic and industry projects. Used as a stepping stone, these software packages are the foundation to more advanced and sophisticated software systems. Recently, there are systems that perform subjectivity analysis at sentence granularity [14]. These systems are capable of understanding consumer sentiment about various types of products in order allocate production budgets. In another situation, a monitoring service (e.g. Nielsen) might be interested in determining relative preferences of TV Shows (a topic that receives substantial mention frequency on Twitter). In essence, the more powerful and fundamental motivation for the system arises in the need to establish preference orderings in order to guide decision-making processes. When companies and organizations can understand and account for consumer sentiment, the result is a net positive outcome for both producers and consumers.

3.2 Sentiment-Based Time Series Analysis on Twitter Data

Combining techniques and models built for doing opinion mining on online text and time series analysis of online topics has produced large success in doing prediction. A study conducted by Pang, et.al discovers high correlation between sentiment word frequency and trends of political opinion [17]. The study even points out the potential of using online text streams as a substitute and supplement for traditional polling. It uses a straightforward approach of counting sentiment word frequency to assess emotional polarity using simple time series techniques. Twitter-specific researches have shown that Tweet-specific attributes like ‘hashtags’ and emoticons can be used to add additional information to sentiment analysis [20, 22]. A group of researchers have incorporated emotions and the AR-ARCH time series model into their analytic system to capture emotional movements on holidays and the occurrence of major disasters [20]. Although researches have shown difficulties and complexities in using hashtags, with appropriate classification techniques, the benefits can outweigh the disadvantages [27].

3.3 Lexical Scoring for Sentiment Analysis

Current system implementation leverages numerous great ideas from the lists of literature above but at the same time evolves from these ideas. Due to the length limit on tweets, each word in a tweet carries great meaning for the overall text. As a result and discussed in later section, the ranking algorithm in the system adopts a word scoring mechanism called SentiWordNet [3]. SentiWordNet is the result of the automatic annotation of all the synsets of WordNet [8] according to the notions of positivity, negativity, and objectivity.

In summary, a wide array of research has been devoted to analyzing online text with time series techniques for predicting real world events. However, no research has addressed the problem of predicting a rank on product sales. This system endeavors to forecast New York Times Top 10 Book Sellers using Twitter data.

4. SYSTEM MODEL

As detailed in Figure 1, the basic overview of the list predictor consists of three components. The Collector is the first component and receives data from Twitter for delivery to the Filter. The Filter decides which books, if any, an incoming tweet refers to and places it in the appropriate “bucket” or discards it as inappropriate. The final component, the Ranker, computes a normalized “sentiment score” for each book in the universe (from prior New York Times lists) based on the sentiment and other information contained in the previous weeks’ worth of tweets in the bucket. This score is independent of any other book. Since these scores are absolute and do not depend on the scores for any

1 An ad-hoc self-assigned categorization (or more generally, tag) of a tweet, prefixed with a hash-mark (e.g. #penn)
other book, the predicted book ranking is simply the universe of books sorted by score.

Each component is detailed more specifically below.

![Figure 1: Overview of system](image)

### 4.1 Collector

The Collector is the component that receives the tweets from Twitter and feeds them for the rest of the system. However, an important secondary function is to construct a corpus because the terms of the Twitter API do not allow any collection of tweet text to be distributed, so none are available [25]. The API terms do permit collections of “tweet IDs” to be distributed but this is not sufficient because Twitter rate-limits lookups quite extensively and there is no longer any mechanism for requesting a rate-unlimited “whitelisted” account.

The overall design of the tweet Collector is a long-running process that receives the data from the tweet stream, does some preliminary filtering (for example, filter non-United States tweets) and parses out the relevant information for the rest of the system. The incoming data feed is extremely verbose — not only is there a great deal of entirely unnecessary information (such as preferred profile background color and profile image URL), but much of the necessary information is verbosely encoded, specifically timestamps and tweet IDs, which are encoded as text. In order to construct a corpus, it is necessary to store months worth of tweets, but at a rate of approximately 10GB per day (based on testing), the raw data stream is clearly too verbose to store directly more than a few weeks’ worth of data, so encoding is necessary.

The Twitter data source that the Collector connects to is the “sample feed”, which is a random sample of all current tweets. This is the most data available without a business relationship and substantial amounts of bandwidth, neither of which are feasible. Using this random sample presupposes that the data required is dense enough for substantial amounts of data to appear in a weeks’ worth of tweets, and also that tweets containing this data are randomly distributed.

### 4.2 Filter

The Filter is the component that divides the incoming tweet stream among a sequence of “buckets”, one per book. Most tweets do not refer to any books, and must be discarded. This means that the data is rather sparse as shown by the compaction factor in the diagram and discussed in the results section below. The Filter is divided in the pipeline into 2 stages. The Primary Filter, which simply looks for general terms in the tweets (such as “read”, “author”, “book”, etc), and the Secondary Filter, which looks for more specific New York Times Bestselling terms (such as author name, book title, etc).

Two different techniques were tried for filtering:

- Filter by a regular expression of a list of words. A tweet passed through the Filter if it contained one of the keywords in the list.
- Filter by reverse word frequency. A tweet passed through if it contained a word that occurred with a significantly higher frequency in a book description than in the English language.

As described the results section, it was found that the first approach produced better results.

The Filter also required a list of books in the universe. For experimental purposes, this consisted of the previous week’s New York Times Best Seller List.

### 4.3 Ranker

Following right after data aggregation and data cleansing comes the analytics module in the pipeline. The Ranker is the software component that is responsible for generating the prediction of New York Times rank trend. The Ranker consumes the filtered tweets from the Filter and runs sentiment analysis algorithms on them to determine polarity (positive or negative feelings expressed in tweets) and intensity (how strong the expressed feelings were). Based on the analysis results, the predictor module outputs a trend prediction for the book for the coming week. The following sections describes each module in detail.
4.3.1 Preprocessor

The preprocessor converts filtered data into usable format for sentiment analysis. Unlike the highly edited genres that conventional natural language processing tools are developed for, tweets contain nonstandard lexical features and syntactic patterns. The goal of the preprocessor is to account for these traits and eliminate, as much as possible, the negative effects on the natural language processing algorithms.

The more relaxed and casual style of conversational text on Twitter raises difficulties such as informal grammar and careless spelling. To address these issues, the preprocessor performs spell correction on tweets. Furthermore, the preprocessor lemmatizes tokens into its base form to make analysis easier later on. In addition, the preprocessor extracts syntactic information from the input tweets. The preprocessor uses a statistical part-of-speech tagger trained on Twitter data[15] to classify tokens to identify different functional tokens. Effectively, the tagger is capable of handling Twitter-specific tags such as hashtags and handles. With the tag information, the preprocessor reduces hashtags and handles into its base form. For example, #Obama is reduced to Obama by the algorithm. After preprocessing, the module hands the data to the sentiment analyzer.

4.3.2 Sentiment Analyzer

The sentiment analyzer analyzes twitter data and computes sentiment scores for the predictor. Not surprisingly, the most important indicators of sentiments are sentiment words. The sentiment analyzer relies heavily on sentiment lexicons to identify positive and negative cues. Cues are partial evidence of valence.

The analyzer does not account for sentiment shifters such as negation or stylistic traits such as sarcasm. The reason is that Twitter postings have a length limit (at most 140 characters) and the authors are usually straight to the point[10]. Additionally, sentiment shifters usually are application-domain specific and thus not suitable for a generalized system. Usage of syntactic patterns has shown decent success in Twitter sentiment classification[7]. Instead of incorporating syntactic patterns, the sentiment analyzer uses syntactic information indirectly to compute sentiment scores.

The sentiment analysis and opinion mining algorithms are detailed in section 4.4.

4.3.3 Predictor

Using the numerical features generated by the sentiment analyzer, the predictor applies Support Vector Machine (SVM) to predict the book’s rank movement for next week. The rank prediction problem is formulated as a classification problem where the possible labels are upward trend, downward trend, or neutral.

In order for the SVM procedure to have a low testing error, it needs to parse features that have the most effect on classification. The most relevant features were the features that had the highest correlation with the upward, downward, and neutral trends. The correlation coefficient was calculated using Pearson’s correlation formula:

\[
r = \frac{\sum x_i y_i - n \bar{x} \bar{y}}{(n - 1)S_x S_y}
\]

(1)

The \(x\)'s represent the value of the features and the \(y\)'s represent the actual ranking. \(n\) is the total number of data points. \(S_x\) and \(S_y\) are the standard deviations of both \(x\) and \(y\) respectively. It should be noted that the above formula only calculates the linear correlation between 2 variables. This limitation does not affect the feature selection much, since features only serve as weak learners, so a general trend is sufficient. For more information on correlation and what it means, refer to [18].

4.4 Sentiment Analysis and Opinion Mining

This section describes the methods the sentiment analyzer uses to detect subjectivity in Twitter postings. There are four unsupervised learning algorithms: Big Dictionary, Emotion Analysis, Emotion Orientation, and SentiWordNet Score. Ultimately, these sentiment indicators are used as features in the predictor to forecast rank movement for the upcoming week.

4.4.1 Big Dictionary

Work in the field of sentiment analysis and opinion mining has produced many lexicons related to affect and emotion. Big Dictionary is an aggregated sentiment lexicon that combines the following lexical resources: Bing Liu’s Opinion Lexicon[9], MPQA [26], WordNet Affect [21], and a hand-compiled emoticon dictionary. An emoticon is a pictorial representation of a facial expression using punctuation marks, numbers and letters. Emoticons are often seen in Twitter and have shown to be an effective feature for emotion classification[19]. Big Dictionary also handles emotions as significant signals for emotion. Big Dictionary serves as a reference to look up whether a token of interest is positive or negative given the syntactic information of the token. If a token is not in the Big Dictionary, the dictionary declares the token to be neutral and has no impact to the overall polarity of the text.

The Big Dictionary Algorithm is a counting-based algorithm. Initialize three counts \(pos\), \(neg\), and \(neu\) to be 0. The algorithm then goes over all tokens in a tweet, lookup on in Big Dictionary and updates the counts accordingly. The bin with the highest count indicates the valence of that tweet.

4.4.2 Emotion Analysis

Beyond subjectivity classification (subjective or not) and polarity recognition (positive or negative), Emotion Analysis algorithm endeavors to analyze tweets with finer granularity and more dimensions. Psychologists have proposed that humans have five basic emotional states: anger, disgust, fear, joy, sadness, and surprise. The algorithm is intuitive and straightforward. Each token in the tweet is labeled with an emotion label by referencing the WordNetAffect dictionary[21]. The emotion class that has the highest count is selected as the label for emotion analysis.

4.4.3 Emotion Orientation

Previous study proposed an unsupervised learning algorithm for classifying reviews as recommended or not recommended using pointwise mutual information[24]. First a part of speech tagger is applied to the review and bigrams are extracted based on predetermined syntactic patterns. Then, the algorithm computes the sentiment orientation:

\[
SO(phrase) = \log \left( \frac{C(phrase, NEAR 'excellent') C('poor')} {C(phrase, NEAR 'poor') C('excellent')} \right)
\]

(2)
where \( C(query) \) be the number of counts returned, given the query \( query \), and \( NEAR \) is the operator that constrains the search within tend words of one another. The reference words 'excellent' and 'poor' were chosen based on these words' correlation with positive and negative reviews respectively. The output of the algorithm is a numerical rating, known as sentiment orientation. Sentiment orientation indicates polarity (positive if \( \text{recommended} \) and negative if \( \text{not recommended} \)) and intensity (based on the magnitude of that number).

The algorithm used in the system borrows the idea of using pointwise mutual information and generalizes to arbitrary emotion classes. The idea is to automatically learn to improve the recognition of emotions. Throughout the paper, the generalized sentiment orientation will be referred to as emotional orientation (EO).

In calculating emotional orientation, denote the tokens in the tweets by \( T = \{ t_1, t_2, \ldots, t_n \} \) and the set of emotion categories by \( E = \{ e_1, e_2, \ldots, e_m \} \). Then the mutual information is computed by

\[
MI(t_i, e_j) = \log \left( \frac{P(e_j | t_i)}{P(e_j)} \right)
\]  

(3)

where \( P(e_j | t_i) \) is the posterior probability that a tweet containing token \( t_i \) implies emotion category \( e_j \) and \( P(e_j) \) denotes the prior probability of the emotion category. Note that if \( P(e_j | t_i) > P(e_j) \) then \( MI(t_i, e_j) \) is positive. Contrarily, if \( P(e_j | t_i) < P(e_j) \) then \( MI(t_i, e_j) \) is negative. The emotional orientation of a token for an emotion category is defined as

\[
EO(t_i) = \sum_{j=1}^{m} P(e_m | t_i) MI(t_i, e_m)
\]  

(4)

When there’s only two classes, computation is done by

\[
EO(t_i) = MI(t_i, E_1) - MI(t_i, E_2)
\]  

(5)

Note that the system lets \( E_1 \) = positive and \( E_2 = \) negative so that emotional orientation is positive when the token is more likely to be associated with the positive emotion.

### 4.4.4 SentiWordNet

Computing a numerical score for individual tweets is one of the integral functions of Ranker. Ranker adopts SentiWordNet, a lexical resource for word-granularity objectivity scoring. The package is designed primarily for sentiment analysis and opinion mining. It is an extension to the well known lexical database WordNet. SentiWordNet assigns to each synset of WordNet three sentiment scores: positivity, negativity, and objectivity. The system gives the system a convenient way to quantitatively evaluate tweets. SentiWordNet is organized as a flat file containing all the scorings for each synset; therefore, to provide programatic access, a module called SentiWordNetCorpusReader serves as an interface between Ranker and SentiWordNet. To query a score for a word, the function call also requires a Part-of-Speech (POS) tag. Therefore, all the preprocessed information is given to the module, including POS tags, to compute a composite score for the tweet. Formally, let tweet \( T = t_1 t_2 \ldots t_n \), where \( t_i \) is the \( i \)th token in the tweet. Each token \( t_i \) has three scores: positive score \( \text{pos}_i \), negative score \( \text{neg}_i \), and objective score \( \text{obj}_i \). Note that the scores are normalized so that \( \text{pos}_i + \text{neg}_i + \text{obj}_i = 1 \).

5. SYSTEM IMPLEMENTATION

The system is implemented in Python. Performance was not a significant concern; however due to the system’s modularity, future applications could easily have performance-critical bottlenecks rewritten in a different language.

### 5.1 Collector

The Collector is implemented in Python and makes use of the PyCURL library to manage the long-lived HTTP streaming connection to the Twitter sample feed. The JSON-encoded data from this feed is decoded using a library and the relevant features of the encoding are extracted and saved in a flat file datastore, split by day. At the moment, the features extracted are tweet ID (a long integer), the timestamp (in RFC 2822 format, plus timezone), and Unicode text.

Since this process must run nearly continuously, there are significant high-availability concerns. We developed a monitoring system to ensure that the data feed connection is maintained, that the Collector process itself continues to run, and that the machine remains available. The monitoring system also ensures that the Collector is restarted should the Twitter server disconnect it, which happens periodically.

### 5.2 Filter

The Filter operates by reducing the massive Twitter dataset collected in stages and achieving finer granularity as it makes progress. This ‘stripping’ procedure ensures that the system begins by eliminating only instances guaranteed to not have objective relevance to the problem domain and steadily increasing the strictness of its targeting as the filtered set decreases in size. This system required only a two-stage filter, partly due to the nature of literary mention frequency on Twitter. Specifically, literature mentions on Twitter are in the first place sparse and, more secondly, relatively easy to detect. The first stage of the Filter therefore accepts an input dictionary of words that are likely to occur in tweets relevant to literature or the act of reading (e.g. “read”, “reading”, “author”). Note that this input dictionary is easily substituted in the event that an alternate problem domain is to be assessed. In practice, the application of the system’s first filtration stage had the profound effect of compacting the Twitter sample set from more than 350 million tweets to approximately 1.5 million. Despite being an unexpectedly constrained result, this had the desirable effect of compacting the set to a size that could easily be handled by filtration at a lower level of granularity, which in this case involved detecting author or title mentions for the novels on books occurring on the New York Times list from the previous week. Detecting these mentions constituted the second stage of our filtration and yielded a sufficiently small set (approximately 15,000 tweets) such that analysis could be conducted efficiently. Overall, it is worth noting that the Filter benefitted from simplicity in its logic over sophistication.

### 5.3 Ranker

Ranker is architectured to be a highly modular software. Python is the programming language of choice for specifying the high-level algorithms and the system takes advantages of software packages that are implemented in lower
level languages. The approach has been effective in prototyping the system as well as achieving performance benchmarks. Also, to account for the amount of data consumed by the pipeline, the system distributes computation tasks across multiple machines whenever possible (no side effect). Preprocessor and the sentiment analyzer have benefited significantly from the distributed computing paradigm. This section discusses design decisions and architectures for each submodule.

5.3.1 Preprocessor

Filter leaves relevant tweets to each book in their respective bins (as a flat file). Preprocessor reads in the file and begins processing. The preprocessor first applies the Twitter POS tagger [15] which is trained with real Twitter data and has an expanded set of POS tags for fine-grained processing. The reason why the system tags tweets before it lemmatizes or spell corrects the function is because the tagger is trained with uncorrected tweets. Processing the tweets before applying the POS tagger in fact loses syntactic information or makes mistakes which then propagates the error forward. The spellchecker is an online package[13] integrated to the system. It uses a very similar model to estimate probability. The reason it is adopted is that the system only corrects a token if the probability of the token being incorrect is high. Otherwise, a high correction rate adds false positive counts to all the counting based sentiment analysis algorithms. Last but not the least, the corrected tweets are then lemmatized with the lemmatizer bundled in NLTK[4].

5.3.2 Sentiment Analyzer

The tagger analyzes the sentence structure and determines the POS for each word. Twitter POS Tagger bases its tagset on the Penn Treebank tagset [12], which is a rich and informative classification system. In addition, tags that are usually social-media-specific such as handles, hashtags, emoticons, and URLs are also indicated by the tagger. Therefore, the analyzer can easily reference the sentiment dictionary given a pair (token, POS tag). Since the tagger and various dictionaries have different tag sets, the system has a mapping that projects return tags from the tagger to a legal POS tag used by the respective dictionary. The sentiment analyzer depends on the numpy package for efficient numerical computation and NLTK for various string manipulation tasks [4, 1].

5.3.3 Predictor

For training a predictor, the system uses libsvm[6] to build a multi-class SVM classifier. The classifier’s task is to predict the rank movement of a book for the upcoming week. The system uses Pearson’s[18] correlation for feature selection. The baseline competition model the system is comparing against is the random guessing model. Later, the system is also evaluated against the majority-classifier model, which always outputs the majority label in the training set. Two training strategies are employed for the study:

- Weighted SVM
- Training via adjusted sampling

Radial basis function kernel is chosen for both models. The standard 10-fold cross validation is applied for evaluation. The motivation for a model comparison is that the datapoint distribution is uneven. There are more downward trend data points than upward trend ones. Thus, a model comparison gives insights into learning the model instead of overfitting the data set.

6. RESULTS

6.1 Collector

The Collector had a number of requirements; in order to construct the very large dataset required for analysis, significant space savings had to be achieved as well as reliability assured so that no data was lost. The Collector was able to meet all these goals with a total uptime of greater than 99.9% over the approximately four months collected, and a total storage of more than 350 million tweets in approximately 30GB. The Collector proved to be a highly reliable component of the system, and the reliability monitoring components performed well. A notable contribution of the Collector is a large Twitter dataset that could be used for future projects (though not distributed, according to the terms of the Twitter API usage agreement)[25]. Furthermore, the Collector code itself can be used to build an arbitrarily-large dataset where required. As the Collector was constrained solely by the Twitter data stream, it should work well for substantially higher-volume streams.

6.2 Filter

After running the Filter on the sample of >350 million tweets, it became clear that literary mentions are not the most popular subject among Twitter users. After running the Primary Filter, only 1 million tweets remained. These tweets referred to any kind of activity which relates to reading or books. After running the Secondary Filter, only about 10,000 tweets remained that related to a New York Times Best Selling book. Even among these, there was some incidence of false positives, since some of the tweets were ambiguous as to whether they were referring to a novel or to the movie version of that novel (“Life of Pi” was one such book). Despite the relatively straightforward method of filtering literature-related tweets, it is hard to imagine an improved version extracting more than four times the volume obtained; this is not enough to make a practical difference to the results, which would require several orders of magnitude more tweets in order to change fundamentally. Nevertheless, 10,000 tweets were enough to make some basic predictions with the Ranker, as discussed in Section 6.

Another approach was made using two versions of the reverse frequency Filter. One of them calculated the frequency of words in a book by parsing Wikipedia articles about the book, and the other by scraping Amazon Customer Reviews about the book. Even though this filter was initially very promising, it ended up having a very high false positive rate. Thus, it was abandoned as a filtering approach.

6.3 Ranker

6.3.1 Feature Selection

The following is a list of features that have a correlation of at least 10% with the ranking direction. This means that they classify as weak learners for predicting the movement of books on the New York Times Bestsellers’ List. Here are the actual features along with their correlations:
An interesting thing to note is that negative emotions, such as sadness cues and negative cues, are much more highly correlated with ranking than positive emotions, such as joy cues. A theory as to why this happens is that people are more likely to post on Twitter if they feel very angry and outraged about a subject. This means that in this case, emotional intensity matters.

Another thing to note is that the correlation with Avg. SentiWordNet and PointWise Mutual Information is actually negative. The reason is that they can take on negative values for negative emotions. Thus, because negative emotions correlate so highly with upward trends, these two features are pulled to the negative side.

6.3.2 Classification Results

The dataset consists of the 154 data points from the top 20 most tweeted books during the period of investigation. For the weighted SVM, all weights $> 2$ yields the same results, suggesting a quick stabilization for the model. The adjusted sampling approach randomly samples downward trend datapoints so that the number of upward points and downward points is the same. Each model is tested 20 times and the average accuracy is reported. The table below summarizes the results of the two competing models and the two evaluation models:

<table>
<thead>
<tr>
<th>Model</th>
<th>Avg. Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random guess</td>
<td>33.33%</td>
</tr>
<tr>
<td>Majority-vote</td>
<td>55.00%</td>
</tr>
<tr>
<td>Weighted SVM</td>
<td>58.37%</td>
</tr>
<tr>
<td>Adjusted Sampling</td>
<td>40.52%</td>
</tr>
</tbody>
</table>

Table 2: Classification results

From the table, it is evident that the weighted SVM achieves the highest accuracy. Majority-vote also has good accuracy rate, beating random guess and adjusted sampling by almost 15%.

Unfortunately, because of the paucity of tweets that referenced New York Times Bestsellers List books, this was the best accuracy achieved. However, as shown in previous work, similar systems can outperform experts on predicting box office returns [2], and perform significantly well in stock market prediction [5].

7. FUTURE WORK

As noted before, the system pipeline constructed is highly extensible and abstractable. Relatively little modification is needed in order to make it applicable to other domains in addition to predicting trends on the New York Times Bestsellers’ List. For instance, a domain of potential interest to clients might be to find trends in the popularity of certain consumer products (such as different brands of sodas). Knowing what the popularity of a product is likely to be could help producers target their commercials accordingly. This modification could be automated by adding an API to the system, which would allow a user to filter the tweets relevant to his or her needs. No modification would be required of the Collector and little modification would be required of the Ranker and Filter.

Additionally, depending on the domain being analyzed, more data sources can be considered besides Twitter (Amazon, Wikipedia, etc). The Collector would have to be slightly modified to account for these new data sources. The end result of this would be an open source application for parsing sentiment of various data sources on various subjects.

As a final side note, it is an interesting aspect of the New York Times Bestsellers List that it can be interpreted on a purely statistical model. This is because most of the books that are on the list on week $x$ are likely to be on the list on week $x + 1$. Since this effort was purely focused on analyzing Twitter data to predict trends, not enough consideration was given to the statistical model. However, it is promising enough that it might considerably increase the accuracy of the Ranker.

8. ETHICS

Given that a large part of this system consists of parsing data on a social network (Twitter), some ethical matters are inherent in the information contained in the social network. If this system were given out as an open source platform with which everyone can analyze tweets, it could be misused to gain private information on Twitter users without their consent. A malicious user could potentially try to extract private information from both individual users, or from particular demographics of people who use Twitter. However it is believed that the ethical implications are limited as Twitter is a broadcast medium; furthermore, only the pre-existing free tweet stream is utilized in the system.

Another matter of concern could be the unwitting promotion of Twitter spam, intended to promote a product. For instance, a company that wants to boost sales of its product might post many Tweets that talk positively about the product, knowing that these would be detected by the system. This process could be automated and might lead to most tweets about the system’s ranked product being machine generated, rather than written by humans. This, of course, would defeat the point of trying to analyze human sentiment on Twitter and lead to increased system load for the Twitter servers.

9. CONCLUSIONS

Given only fourteen weeks of actual data, the system yielded relatively significant results and it did considerably better than random guessing. It is worth noting that it performed well despite a number of severely problematic factors, each of which alone could have been crippling. In particular: the collected dataset spanned only a short interval of time when several years worth would have been preferable, literary mention frequency on Twitter is surprisingly sparse, and it proved quite difficult to remove false positives via filtration due to fascinating edge cases that would have only been detectable via sophisticated linguistic analysis. To re-
iterate: despite being hamstrung by these various obstacles, the system has proven remarkably effective in conducting a relational favorability analysis on bestselling fiction literature.

Critically, the system delivered satisfies the core objectives of collecting, filtering and ranking a Twitter data stream. It is architectured in a pipeline fashion such that the analysis can be performed as batch jobs on a variety of chosen domains. As noted, it can easily be abstracted to other topic domains as desired. It is also extensible: given a desire to assess other features of a particular topic domain, such an analysis can be appended to a component of the Ranker. Lastly, the system is scalable because its methodology is fundamentally parallelizable.

The application of this system towards the topic domain of popular fictional literature served to calibrate it for execution in future analyses. It has been structured as ‘black box’ software such that alternative domain analyses are achieved simply by altering a small set of inputs. Future work therefore would not only encompass refining it but also seeing it applied to other domains wherein relative predictions might prove useful to decision-making processes.

Conclusively, this system serves to validate the hypothesis that social media sentiment can be correlated with relative favorability across specific topic domains.

10. REFERENCES


[16] Alexander Pak and Patrick Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapia, editors, Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC’10), Valletta, Malta, may 2010, European Language Resources Association (ELRA).


[18] Karl Pearson and Louis Napoleon George Filon. Mathematical contributions to the theory of evolution. iv. on the probable errors of frequency constants and on the influence of random selection on variation and correlation. Philosophical Transactions of the Royal


