Expert Systems with Applications 40 (2013) 6266–6282

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Twitter brand sentiment analysis: A hybrid system using *n*-gram analysis and dynamic artificial neural network

M. Ghiassi [a,*], J. Skinner [b], D. Zimbra [a]

[a] Santa Clara University, Santa Clara, CA 95053, United States
[b] Amazon.com, Inc., Seattle, WA 98109, United States

ARTICLE INFO

Keywords:
Twitter
Sentiment analysis
Twitter-specific lexicon
DAN2
Feature engineering
n-gram analysis
Machine learning
SVM

ABSTRACT

Twitter messages are increasingly used to determine consumer sentiment towards a brand. The existing literature on Twitter sentiment analysis uses various feature sets and methods, many of which are adapted from more traditional text classification problems. In this research, we introduce an approach to supervised feature reduction using *n*-grams and statistical analysis to develop a Twitter-specific lexicon for sentiment analysis. We augment this reduced Twitter-specific lexicon with brand-specific terms for brand-related tweets. We show that the reduced lexicon set, while significantly smaller (only 187 features), reduces modeling complexity, maintains a high degree of coverage over our Twitter corpus, and yields improved sentiment classification accuracy. To demonstrate the effectiveness of the devised Twitter-specific lexicon compared to a traditional sentiment lexicon, we develop comparable sentiment classification models using SVM. We show that the Twitter-specific lexicon is significantly more effective in terms of classification recall and accuracy metrics. We then develop sentiment classification models using the Twitter-specific lexicon and the DAN2 machine learning approach, which has demonstrated success in other text classification problems. We show that DAN2 produces more accurate sentiment classification results than SVM while using the same Twitter-specific lexicon.

1. Background

Twitter offers a unique dataset in the world of brand sentiment. Public figures and brands receive sentiment messages directly from consumers in real time in a public forum. Both the targeted and competing brands have the opportunity to dissect these messages to determine changes in consumer sentiment. Taking advantage of this data, however, requires researchers to deal with analyzing an immense amount of data produced by Twitter each day, referred to as the Twitter fire hose. As noted by Twitter employee Taylor Singletary on the Twitter.com developer discussion site, "Firehose access is very hard to come by and potentially very expensive to realistically consume." The most prolific account on Twitter receives over 300,000 Twitter messages a day (e.g. Justin Bieber). Services that allow marketers to analyze a dataset of this magnitude (NetBase Solutions, Lexalytics, Converseon, Summize (Jansen, Zhang, Sobel, & Chowdury, 2009)) are becoming increasingly important in the world of brand management. Modeling and analyzing such large datasets often introduce a large number of variables or features. Go, Bhayani, and Huang (2009) report using a unigram model with more than 40,000 features for Twitter sentiment analysis. Twitter's own Summize (Jansen et al., 2009) uses a lexicon of 200,000 unigrams and bi-grams. These models use a large set of features but have resulted in low prediction accuracy of tweet sentiment.

The language found in Twitter messages has often stymied researchers' ability to determine sentiment. Using term pairs and a mutual information approach, Jansen et al. (2009) found that "more than 80% of the tweets that mentioned one of these brands expressed no sentiment." Bollen, Pepe, and Mao (2011) dropped 88.6% of all tweets gathered and reported, "we only retain tweets that match the following regular expressions "feel", "I'm", "Im", "am", "^am", "being", and "be"." Pak and Paroubek (2010) *n*-gram approach was intolerant of conflicting emoticons, "we assume that an emoticon within a message represents an emotion for the whole message and all the words of the message are related to this emotion". And in a separate project involving distance supervision, Go et al. (2009) had to completely remove emoticons from their dataset, because they found that "if we leave the emoticons in, there is a negative impact on the accuracies of the MaxEnt and SVM classifiers".

Many researchers have experienced challenges in collecting data using the Twitter API (the interface used to access public Twitter data). Go et al. (2009) reported that "The Twitter API has a limit of 100 tweets in a response for any request". This limited their data set to 100 records every 2 min and resulted in a non-continuous

* Corresponding author. Tel.: +1 408 554 4687; fax: +1 408 554 2331.
E-mail address: mghiassi@scu.edu (M. Ghiassi).

0957-4174/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2013.05.057

collection. For the purposes of monitoring brand sentiment, such discontinuity may result in important communications missed. Many prior studies collected data using the Twitter API v1.0, as we do in this study. Their collections were limited in size by the technical challenges posed by the API, and their devised approaches were unable to truly perform continuous monitoring of Twitter sentiment. In this study, we overcome these challenges in using the Twitter API v1.0 and develop tools to collect a large and continuous Twitter corpus. Many of the data collection challenges that we addressed using the Twitter API v1.0 were alleviated in v1.1, which was released after our data collection period, in September 2012.

Some researchers have limited the amount of data in their investigations as well: Pak and Paroubek (2010) analyzed 300,000 records, but employed a training set of only 216 records. Jansen et al. (2009) analyzed 149,472 tweets, but relied on a third party service, Summize, to determine sentiment, which as the authors noted "Summize uses only the most recent 125 tweets for the time period to determine overall sentiment". It should be noted that Jansen et al. (2009) manually coded 2610 tweets to determine if the Summize service was accurate. They found that "Using the general linear model (Dobson 2002), we identified no difference between the sentiment distributions between the first 125 and second 125 tweets by brand. Therefore we are reasonably assured that the most recent 125 tweets are a representative sample of the tweets for given weeks." Finally, Go et al. (2009) manually marked 359 tweets for sentiment.

Rui, Liu, and Whinston (2011) summarized the problem best in their conclusion, "On the one hand we are happy to see large volumes of WOM (*word of mouth*) data because it reduces the sample bias; on the other hand, analyzing people's attitudes becomes a challenge because manually checking each WOM message is obviously not feasible."

We identify four distinct problem areas associating with measuring Twitter sentiment that need further investigation: data gathering, determination of a sentiment scale to apply to the data, feature engineering, and evaluation and classification of the Twitter messages. The following sections will discuss the considerations associated with each problem area.

The remainder of this paper is as follows: Section 2 presents a summary of the existing Twitter modeling approaches and introduces the approach used in this paper. Section 3 discusses data gathering and Twitter-specific sentiment topics. In Section 4 we present our feature engineering approach and introduce how we tailor feature engineering tools for feature construction, reduction, selection, and customizations. Section 5 discusses how we employ DAN2 and SVM for automated sentiment analysis and present the results. We then compare performance of these methods for our dataset. This section also discusses the effectiveness of the Twitter-specific lexicon by comparing its performance against a traditionally produced lexicon set. Finally, Section 6 presents our conclusions.

2. Twitter sentiment analysis literature review and modeling approach

Twitter is a popular and rapidly growing computer-mediated communication platform. Twitter users create micro-blog status update messages called tweets to communicate with other users for various reasons on a wide variety of topics. These tweets often contain valuable information and the perspectives and opinions of users on issues related to business and society (Gleason, 2013; Jansen et al., 2009). Researchers have developed various approaches to monitor Twitter in real-time for the occurrences of major events, the outbreak of news stories, and the reactions of the users to events (Benhardus & Kalita, 2013; Bifet & Frank, 2010; Hsieh, Moghbel, Fang, & Cho, 2013; Mathioudakis & Koudas, 2010; Naveed, Gottron, Kunegis, & Alhadi, 2011; Petrovic, Osborne, & Lavrenko, 2010; Phelan, McCarthy, & Smyth, 2009; Thelwall, Buckley, & Paltoglou, 2011). Researchers have also studied the dynamics of the diffusion of information throughout the Twitter user population (Lerman & Ghosh, 2010; Romero, Meeder, & Kleinberg, 2011). Social network analysis researchers have examined the influence of users in the Twitter network (Bakshy, Hofman, Mason, & Watts, 2011; Cha, Haddadi, Benevenuto, & Gummadi, 2010). Twitter has also been used by researchers as a source of information to explain various major events or indicators of business and society, demonstrating the value of these communications. For example, researchers have explained and predicted the outcomes of major political elections through the analysis of tweets on the candidates and political issues (Bermingham & Smeaton, 2011; Chung & Mustafaraj, 2011; Mejova, Srinivasan, & Boynton, 2013; O'Connor et al., 2010; Ringsquandl & Petkovic, 2013; Tumasjan, Sprenger, Sandner, & Welpe, 2010; Wang, Can, Kazemzadeh, Bar, & Narayanan, 2012). Researchers have also utilized Twitter as a source of information to explain and predict movements in stock markets and other socioeconomic indicators (Bollen, Mao, & Zeng, 2011; Bollen, Pepe, & Mao, 2011Bollen, Pepe, & Mao, 2011; Mittal & Goel, 2012). The majority of studies in these areas of Twitter research employ sentiment analysis approaches to identify and evaluate the opinions of users expressed in their tweets.

Sentiment analysis is a prominent and active area of research, spurred particularly by the rapid growth of web social media and the opportunity to access the valuable opinions of numerous participants on various business and social issues. Sentiment analysis has been performed in a wide variety of genres of communication, including news articles (Tetlock, 2007), product reviews (Dave, Lawrence, & Pennock, 2003; Gamon, 2004; Pang, Lee, & Vaithyanathan, 2002), and web forums (Abbasi, Chen, & Salem, 2008; Das & Chen, 2007). Approaches to sentiment analysis aim to identify and evaluate the opinions expressed in writing, a task considered orthogonal to topical analysis.

Techniques for sentiment analysis can be broadly categorized into two classes of approaches. The first class involves the application of a sentiment lexicon of opinion-related positive or negative terms to evaluate text in an unsupervised fashion (Turney, 2002). The second class of approaches utilize a textual feature representation coupled with machine learning algorithms to derive the relationship between features of the text segment and the opinions expressed in the writing in a supervised fashion (Pang et al., 2002). Models based upon such supervised techniques require a large training set of instances complete with class labels, to calibrate the models and tune the relevant parameters. Labels can be assigned to training instances manually through human evaluation of the text, or resources with explicitly defined ratings (such as the number of stars assigned in movie and product reviews) can be leveraged.

Much of the existing research in sentiment analysis examines two major issues: the features utilized to represent the unit of writing (Abbasi et al., 2008), and the techniques applied to perform the sentiment analysis (Pang et al., 2002). Several classes of writing features have been applied in the literature, including semantic (Turney, 2002), syntactic (Pang et al., 2002), and stylistic features (Abbasi et al., 2008). Semantic features include manually or semi-automatically generated sentiment lexicons of specific terms categorized as expressing positive or negative opinion in a particular domain (Das & Chen, 2007; Tetlock, 2007; Turney, 2002). In addition to semantic features, syntactic features are most commonly applied, particularly word *n*-grams and part-of-speech *n*-grams (Dave et al., 2003; Gamon, 2004; Pang et al., 2002). Distinctive writing style is prevalent in web discourse, and stylistic features often incorporated in stylometry studies have also shown

to be effective in sentiment analysis (Abbasi et al., 2008). The inclusion of numerous writing features in the representation of a unit of text often results in an expansive feature space that is typically reduced through feature selection prior to classifier calibration, to identify and apply the most significant discriminators of opinion expression in the domain (Abbasi et al., 2008; Gamon, 2004).

To utilize these writing features in the sentiment analysis, various techniques have been proposed in the literature. Score-based methods are typically used in conjunction with sentiment lexicons to analyze a unit of writing (Kim & Hovy, 2004; Turney, 2002). In the score-based approach the frequencies of occurrences of positive and negative terms in the lexicon are compared to produce an overall sentiment score and classification. Many prominent approaches to sentiment analysis utilize supervised machine learning techniques to develop classifiers (Abbasi et al., 2008; Dave et al., 2003; Gamon, 2004; Pang et al., 2002). Among the various machine learning techniques, support-vector machines and naïve Bayes are most commonly applied (Pang et al., 2002), although artificial neural networks and maximum entropy models have also demonstrated success in sentiment analysis applications.

Another major area of Twitter research focuses on developing sentiment analysis approaches specifically designed for tweets. Tweets are a unique genre of communication, and its unique features and properties have brought into question the applicability and effectiveness of the more traditional approaches to sentiment analysis. Tweets are very short units of text, at maximum 140 characters long, and characterized by casual, compact language with extensive usage of slang, abbreviations, acronyms, and emoticons. Tweets also contain hashtags, user references, and embedded links to other websites containing additional referenced information, further complicating the sentiment analysis.

Twitter sentiment analysis has produced inconsistent results in prior studies, which have caused researchers to question the effectiveness of the analysis and advise caution in drawing conclusions based upon its results (Wong, Sen, & Chiang, 2012). Wong et al. performed sentiment analysis on tweets related to movies, and found these sentiments to be different than those of users of online review sites discussing the same movies. Furthermore, performing sentiment analysis on the tweets was unable to yield predictable indications of box-office returns, limiting the usefulness of the information source. These findings may be due in part to the complexities of performing sentiment analysis in Twitter. The authors in the study utilize a well-adopted approach from the literature describing sentiment analysis on more traditional genres of communication. Applying a word unigram feature representation to Twitter sentiment analysis results in high dimensionality with a large amount of statistical noise in the data, which may explain the inconsistency in their findings. Approaches designed specifically for sentiment analysis in Twitter may produce more meaningful, accurate, and reliable information.

To perform sentiment analysis on tweets, researchers have applied the more traditional approaches that have demonstrated success in other genres of communication, such as news articles or reviews, as well as devised improved methods that leverage the unique features of Twitter. Many researchers have utilized sentiment lexicons developed for traditional genres of communication, such as the Opinion Finder lexicon (Wilson et al., 2005), with score-based methods to perform unsupervised sentiment analysis on tweets (Bermingham & Smeaton, 2010; Bollen et al., 2011; Go et al., 2009; O'Connor et al., 2010; Tumasjan et al., 2010; Bollen, Pepe, & Mao, 2011; Hu, 2013). Researchers have also utilized various textual feature representations of tweets coupled with machine learning algorithms to derive the relationship between features of the tweet and the opinion expressed in a supervised fashion. Similar to the approaches to sentiment analysis applied to other genres of communication, semantic, syntactic, and stylistic textual features have been examined, as well as features that are more specific to tweets. Researchers have utilized semantic measures in the tweet feature representations by incorporating sentiment lexicon terms (Agarwal, Xie, Vovsha, Rambow, & Passonneau, 2011; Barbosa & Feng, 2010; Jiang, Yu, Zhou, Liu, & Zhao, 2011; Kouloumpis, Wilson, & Moore, 2011; Saif, He, & Alani, 2012). Syntactic measures such as word *n*-grams and part-of-speech *n*-grams have also been applied in tweet feature representations (Bermingham & Smeaton, 2010; Davidov, Tsur, & Rappoport, 2010; Go et al., 2009; Huang, Peng, Li, & Lee, 2013; Jiang et al., 2011; Kouloumpis et al., 2011; Mejova et al., 2013; Pak & Paroubek, 2010). Features designed to capture expressions of writing style in tweets have also been examined (Agarwal et al., 2011; Barbosa & Feng, 2010; Davidov et al., 2010; Jiang et al., 2011). Due to the casual, compact language utilized in Twitter, researchers have also evaluated features that are more specific to the domain, including acronyms (Agarwal et al., 2011; Barbosa & Feng, 2010; Kouloumpis et al., 2011) and emoticons (Agarwal et al., 2011; Jiang et al., 2011; Kouloumpis et al., 2011; Liu, Li, & Guo, 2012; Zhang, Ghosh, Dekhil, Hsu, & Liu, 2011). In utilizing these tweet feature representations in the sentiment analysis various machine learning techniques have been evaluated by researchers, although support vector machines (Agarwal et al., 2011; Barbosa & Feng, 2010; Bermingham & Smeaton, 2010; Go et al., 2009; Jiang et al., 2011; Zhang et al., 2011) and naïve Bayes models are most often applied (Go et al., 2009; Bermingham & Smeaton, 2010; Pak & Paroubek, 2010; Saif, He, & Alani, 2011; Saif et al., 2012).

While some researchers have recognized the unique features and properties of the language used in Twitter and attempted to incorporate some of these aspects in their textual feature representations through, for example, the inclusion of acronyms and emoticons, few have addressed this uniqueness directly in a more comprehensive fashion. The majority of studies in the area continue to rely upon semantic features and sentiment lexicons developed for other domains and more traditional genres of communication. Furthermore, efforts to expand these sentiment lexicons to form more comprehensive feature representations for application on tweets have leveraged traditional lexical resources like WordNet (Agarwal et al., 2011). However, sentiment analysis research has shown such features and models are domain-specific, and analytic performance degrades significantly when these are applied in other domains (known as the domain transfer problem) (Aue & Gamon, 2005; Blitzer, Dredze, & Pereira, 2007; Tan, Wu, Tang, & Cheng, 2007). In this research, we present a supervised approach to the development and application of a Twitter-specific sentiment lexicon, recognizing and capturing the unique characteristics of the Twitter language. We extend this Twitter-specific sentiment lexicon to include brand-specific terms, and show an improvement in sentiment analysis performance through the inclusion of such features. Finally, while most studies have applied support vector machines or naïve Bayes models to develop sentiment classifiers, we evaluate the performance of a dynamic artificial neural network model in Twitter sentiment classification, which has demonstrated success in other text classification applications (Ghiassi, Olschimke, Moon, & Arnaudo, 2012).

In this research, we introduce a hybrid approach that uses *n*-gram analysis for feature extraction, and a dynamic artificial neural network (DAN2) (Ghiassi & Saidane, 2005) algorithm or SVM as alternative approaches for Twitter sentiment analysis. We view Twitter sentiment analysis as a text classification problem. The objective in this analysis, therefore, is to classify each tweet into a sentiment class, often three or five. Collectively, tweets' corpus sentiment represents the perspective of the dataset toward a subject. Traditionally, tweets may be analyzed to produce a sentiment score. The scoring process may discriminate among the features of a tweet by assigning different weights to features or by using a

data analysis approach to obtain the weights. Once a tweet is scored, it can be mapped into one of the classes. The sentiment score continuum is partitioned so as to represent the three or five sentiment classes. Once a set of boundary values for each partition is identified, tweets are assigned to the appropriate class. Clearly, identifying a set of weights to achieve consistent sentiment scoring is a modeling challenge. We offer machine learning tools as an alternative solution for this problem.

The sentiment scoring process uses feature engineering methods to first identify relevant terms/features for tweets. This identification uses vocabulary (unigrams) or partial or full semantics (*n*-grams). In this research we use unigrams, bi-grams, and tri-grams for feature identification.

The Twitter corpus feature identification often results in a very large set, numbering in tens of thousands of features. Modeling any system with such a large number of features (variables) is often complex. Therefore, once features of a Twitter corpus are identified additional feature engineering activities can be used to reduce the number of features to a manageable set.

We view the sentiment of a tweet as a function of its features. Most studies use some pre-existing functional form for this representation, such as applying a sentiment lexicon and a score-based method to evaluate a tweet (Bollen et al., 2011; Bollen et al., 2011Bollen, Pepe, & Mao, 2011). In this research we do not assume existence of a specific functional form a priori and use artificial neural network modeling approach for sentiment representation. This approach is data driven in which tweets are presented as vectors of zeros and ones. The Twitter corpus, thus, is represented in vector space for analysis. The generated input matrix for this model is a sparse matrix in which a one represents the presence of the feature in the tweet and a zero its absence. The dependent variable in this model is the class designation of the tweet.

In order to assess the sentiment of a tweet (and the corpus) toward a subject, the model needs to be trained. The training process requires generation of the vector representation for each tweet and the sentiment class it belongs to. Once the feature set is determined, creation of the input matrix is easily achieved. The sentiment class designation for the training dataset, however, is a manual operation. We developed automated tools to assist human evaluators to classify tweets for the training of the model.

Similar to other text classification problems, we will use a training dataset and a separate testing dataset to train the model and to assess its accuracy. To show the effectiveness of this approach we use two classification tools (DAN2 and SVM) and compare their performance using identical datasets.

3. Data collection and preparation

We use the Twitter API v1.0 for our data collection. The API v1.0 offered by the Twitter service is a moving target, and has changed several times even over the course of our investigation. The most common and consistent method for gathering data is to request a paged set of data for a given query. The subject (brand) selected for this research is Justin Bieber. At the time of this research, his Twitter account was the largest Twitter account receiving more than 300,000 tweets daily, eclipsing the next three largest accounts together. In our investigation we developed a set of data gathering tools that both continually monitored the Twitter service for @justinbieber mentions (using a paging query) and also audited periods of time for any missed @justinbieber references. Auditing turned out to be a critical function as the Twitter API was frequently offline during the period of investigation. The data gathering tools were used to capture a total of 10,345,184 tweets from May 6th at 7am to June 8th at 7am.

3.1. Sentiment scale

Literature defines various sentiment scales. Sentiment scales include mood states, tones, and simple positive to negative ranges (-1 to +1, or -100 to +100). The key considerations in selecting a sentiment scale are: (1) the ability to develop an association between message features and steps on the scale, and (2) the ability to develop a mechanism for human supervision of the results.

3.1.1. Relationship between message features and scale steps

It must be possible to relate message features to steps on the sentiment scale. Natural language processing (NLP) models analyze sentence parts that can be valued with different weights, resulting in a fine grained scale. A commercial sentiment analysis product based on this approach (Lexalytics) offers a 200 point scale (from -1.00 to +1.00). Go et al. (2009) on the other hand, employed a three point scale (“positive”, “neutral” and “negative”) that included the following definition for neutral, “If the tweet could ever appear as a newspaper headline or as a sentence in Wikipedia, then it belongs in the neutral class.” The authors also categorize all factual statements as neutral (e.g. “just landed at San Francisco”). The Lexalytics approach, however, is fine grained enough to judge the degree of sentiment that exists in any statement. The Lexalytics wiki states, “We consider neutral to be a range where there might be some measurable sentiment within a piece of content, but not enough that you could firmly state the document is positively or negatively toned.” Bollen et al. (2011) introduced a six point sentiment scale. In their paper “Twitter Mood Predicts the Stock Market” they use the Google-Profile of Mood States (GPOMS) with six mood dimensions (Calm, Alert, Sure, Vital, Kind, and Happy).

Twitter’s own Summize engine uses the five Likert scale classes (wretched, bad, so–so, swell, and great) (Jansen et al., 2009) for its classification. We follow a similar scale and define five sentiment states for our analysis (Strongly Positive, Mildly Positive, Neutral, Mildly Negative, and Strongly Negative). Finally, we acknowledge that the number of steps on the sentiment scale can determine the attempted precision.

3.1.2. Ability to supervise the results

The key problem with a fine grained scale is that the result is difficult to validate against a human-produced score. Even with detailed guidelines, it would be challenging to find two different evaluators that could produce identical 200 point values for a large set of Twitter messages. This problem is even further exacerbated when the scale attempts to depict something more than a negative to positive range (such as mood). Clearly scales with fewer steps are easier to validate (between a programmatic score and a human produced score).

As the steps on the scale are reduced, however, a problem begins to arise with the middle, (neutral) category. Go et al. (2009) gave his evaluators a clear guideline for the neutral category: facts were considered neutral. Their approach, however, does not take into account the promotional impact that is intended when an author broadcasts engagement with a brand (e.g., “Listening to JustinBieber!”). In their model all statements describing the use of a product fall into the neutral category. We consider this to be a flaw, often resulting in a large, coarse neutral bucket; this approach will result in wide sentiment buckets that are easier to supervise, but sacrifice precision.

4. Feature engineering

The Twitter sentiment analysis is a special case of the general category of text classification. Text classification problems are complex in nature and are always characterized by high

dimensionality (Yang & Pedersen, 1997). To reduce this complexity researchers begin by applying preprocessing techniques to the original documents in order to produce a more simplified text.

We use standard preprocessing activities in our feature engineering stage. These are: (1) removing stop words, (2) stemming, (3) transforming the data into the vector space, (4) term weighting, (5) feature selection, and (6) splitting the data into training and test sets. These preprocessing steps transform the documents into a simpler form and a vector representation of the documents is produced (stages 1–4 of standard steps). This vector is called the bag-of-words. Literature on information retrieval lists approaches commonly used for preprocessing activities and the steps necessary to produce such a vector (Frakes & Baeza-Yates, 1992; Manning, Raghavan, & Schütze, 2008). The transformed document's value for each term is defined using various term weighting methods. Common term weighting methods are binary weighting, the term frequency approach (TF), and the term-frequency/inverse document frequency (TF/IDF) method (Manning et al., 2008; Salton & Buckley, 1988). Automated software tools are available to perform some or all of these steps. Even after preprocessing, for large scale documents, the size of the resulting matrix is still very large and further activities are required to reduce its size. Feature engineering methods are introduced to reduce the size of such matrices. In text classification models, a feature represents a variable. Most real life text classification problems, including the Twitter dataset used in this study, often involve 1000s or 10,000s of features. Feature engineering is applied in order to reduce the number of features. This is a common step for classification methods in order to reduce complexity (Yang & Pedersen, 1997). Additionally, feature engineering serves as a means to remove noise from the input documents (Manning et al., 2008). The most widely used feature engineering approaches are the information gain (IG), document frequency thresholding (DF), mutual information (MI), term strength (TS), latent semantic indexing (LSI), and the χ^2 statistic (CHI) (Manning et al., 2008; Yang & Pedersen, 1997). We also use feature engineering to reduce complexity in our analysis. We next examine applicability of traditional feature engineering tools and introduce a new measure specific for Twitter feature selection.

There are specific challenges associated with feature engineering as it pertains to Twitter sentiment analysis. One of the most widely used method for feature selection for text classification systems rely on TF/IDF weighting (Ghiassi et al., 2012; Joachims, 1999). This approach places value on highly discriminating terms in the document corpus. Manning et al. (2008) defines the calculation $TF/IDF_{t,d}$ as an assignment to term t a weight in document d that is:

1. Highest when t occurs many times within a small number of documents (thus lending high discriminating power to those documents);
2. Lower when the term occurs fewer times in a document, or occurs in many documents (thus offering a less pronounced relevance signal);
3. Lowest when the term occurs in virtually all documents.

However, sentiment within the Twitter corpus poses four key challenges that cannot be answered by measures such as TF/IDF. We next discuss these challenges and propose four properties of a feature engineering approach specific to Twitter sentiment analysis.

4.1. Strong sentiment terms are pervasive

First, there are strong sentiment terms that are pervasive throughout the Twitter corpus. Any approach that discounts these widely used terms due to lack of uniqueness will misrepresent the sentiment associated with the Tweets in which they occur.

Table 1 captures word frequency for a variety of key sentiment terms across the 10,345,184 tweets from the Justin Bieber dataset:

With a TF/IDF based weighting approach, the feature "Love" would be underweighted (due to its pervasive use) while the feature "Luv" would be more strongly weighted. The two terms are used interchangeably in the Twitter corpus, however (i.e. they are semantically identical), and thus no discount should be applied to one term over the other (Table 1).

Similarly, the smiley-face emoticon ":)" occurs 4.5 times more often in the left to right orientation ":)" than in the right to left orientation "(:". The two emoticons are semantically identical, with the disparity attributed mainly to personal aesthetics and the orientation difference between Latin and ideographic (Chinese, Korean) scripts. Were a TF/IDF approach to be used to determining the feature weights of the smiley-face variants, it would improperly penalize the widely used ":)" and reward the lesser used "(:" variant (Table 1).

An even more significant variation can be found in the set of sad-face emoticons. The sad-face emoticon ":(" appears 11.3 times more often in the left to right orientation ":(" than in the right to left orientation "):". A reasonable conclusion to this problem would be to combine identical terms, and apply weights to the set of like terms. Combining terms, however, makes the set of terms less unique (than any individual member), more pervasive and results in even a lower TF/IDF weight.

Property 1. In Twitter analysis, the pervasive use of a few strong sentiment terms, and the light use of analogous terms, must be tolerated by the feature construction and feature weighting approach.

4.2. Syntactical context must determine feature boundary

The next key challenge in weighing a set of features for Twitter sentiment classification is the manner in which boundary conditions are determined for each feature. A stemmed query against the corpus for the term "hate" (i.e. "hate%") results in a record set somewhat evenly divided between authors that "hate" Justin Bieber and authors that want to defend Justin from the "haters". This causes "hate%" to be highly indiscriminate when it comes to sentiment. An example of one of these problematic Tweets is as follows:

"@justinbieber Ignore all the haters their just prejudice idiots."

If a feature was allowed to include the full set of Tweets found with the stemmed query ("hate%") that feature would not be able to clearly describe sentiment in one direction or another (due to

Table 1
Frequency of common sentiment terms.

Feature	Frequency (%)	Feature	Frequency (%)	Feature	Frequency (%)
Love (stemmed)	23.65	Hate (stemmed)	1.30	Please	17.28
Luv (stemmed)	0.47	Hate (exact match)	0.53	Please follow	7.76
:)	14.42	Hater (stemmed)	0.60	Follow	34.66
(:	3.22			Follow me	23.03
:(	3.63				

the inclusion of both "hate" and "hater"). Table 1 denotes the frequency of "hate" (the stemmed version), "hate" (the exact match, a better candidate for negative sentiment) and "hater" (the stemmed version, which is an excellent candidate for positive sentiment). The catch-all query for "hate" includes more references to "hater" (0.602% of the corpus) than to "hate" itself (0.529% of the corpus). Thus the stemmed "hate" query cannot be used to create a feature with a meaningful syntactical boundary. The system used for weighing sentiment terms must be able to identify that the stemmed variation of "hate" includes a significant syntax variation and is therefore deficient in identifying sentiment.

Property 2. The ability to determine precise feature boundaries must be included in the feature construction approach.

4.3. The "please follow me" phenomenon – unavoidably broad classes

Another distinct problem with Twitter sentiment categorization is that some of the classes into which messages are categorized are unavoidably large. This is a result of both the way in which Twitter users choose to employ the messaging system, and the nature of sentiment itself. For example, in the case of the Twitter network, a common activity is to expand one's set of followers by asking a high profile individual to "follow me." This activity is so common that over a third of messages (34.7%) directed to Justin Bieber include the term "follow." Table 1 denotes the term frequency for common unigrams and bi-grams associated with follow requests ("please", "please follow", "follow", "follow me").

Term frequency analysis reveals that follow requests are so pervasive on the Twitter network that were a single class defined to hold the requests, it would contain 34.7% of all Tweets. Furthermore, the sentiment term "please" cannot be effectively used to recognize sentiment outside of follow requests because follow requests account for 45% of all uses of the term "please" on the Twitter service (e.g. 7.763/17.281 = 44.92%, Table 1). Put another way, any algorithm that attempts to judge sentiment using the term "please" as a stand-alone feature will pick up "follow" sentiment 45% of the time.

Thus an approach such as TF/IDF that values clear, distinct bursts of a term (or cluster of terms) in a statistically small number of documents (that correspond to a narrow category) will not be able to optimally select features for the wide sentiment buckets that are required to hold Twitter sentiment.

Property 3. The feature weighting approach must be able to weigh the performance of features that are associated with wide sentiment classes. Furthermore, to increase the precision of the wide sentiment classes, the affinity that unigram terms show for one another must be considered at the time of feature engineering.

4.4. Real world bias toward positive sentiment

A key problem with Twitter sentiment is that there is considerably less positive feedback provided to the author after publishing a negative Tweet:

- Fewer re-tweets by the subject – The subject of a negative tweet is, obviously, predisposed against re-tweeting the message.
- Fewer re-tweets by readers – The Twitter user that reads a negative Tweet on a given subject either (1) made a conscious decision to follow the author or (2) made a conscious decision to follow the subject. In the case of a reader following the subject, there is some degree of effort involved in following the full set of messages pertaining to the subject. Thus it is likely that the reader will not further promulgate negative sentiment toward something they have invested time and effort into.
- Fewer follows – Both the subject of the negative tweet, and those disposed positively toward the subject will naturally exhibit a bias against following the author of a negative tweet.
- Wrath of the mob – The more widely beloved a subject is, the more likely the author of a negative message is to be inundated with responses defending the subject.

This results in a corpus that is biased toward positive sentiment. In the 2918 rows randomly selected for this investigation approximately 85.4% were found to have either mild or strong positive sentiment. Lexalytics confirms a positive slant in the full Twitter corpus, though not to the degree we find in the Justin Bieber corpus, "We find that overall document content is slanted to be slightly positive by default". Evaluating the term frequency of sentiment terms in the Twitter corpus with a TF/IDF-based approach would result in a feature set that does not value commonly found, positive terms (as these terms are pervasive throughout the document set, which TF/IDF equates to a less pronounced relevance signal).

Property 4. The unbalanced classes found in the Twitter corpus require the feature engineering approach to normalize for the larger positive sentiment class when measuring term frequency. Unbalanced classes must also be considered when determining the affinity a term has for a sentiment class (a key element of feature boundary engineering, described further).

Our approach will engineer features in a manner that addresses problems described in this section and will rely on a sentiment scale that is both precise and free of interpretation bias as discussed earlier. In the following sections we introduce our approach for Twitter-specific feature engineering and Twitter sentiment analysis.

4.5. Evaluation of Twitter messages

Twitter messages use some specialized terms and are often short. Applying feature engineering techniques to Twitter messages, thus, require special considerations. The first consideration is the supervised training required for such analysis. In order to train a model, humans need to be able to evaluate and produce consistent score for like messages. Similarly, when using *n*-gram analysis the explanatory power of the feature set and the interaction between features must be considered. Our approach attempts to address both considerations, by starting with a large feature set which can also be reduced to a supervised score which maps to a step on a precise sentiment scale in real-time.

Another feature of Twitter messages is the offsetting nature of sentiment terms. Features that describe sentiment offer a distinct problem in text classification in that they can negate one another when found alongside one another in a given message. For example, if a message contains a feature that captures the term "not" and a feature that captures the term "happy", and these features are in close proximity in the message, it is very likely that the author of the message was in fact "not happy" about something. An evaluation method that fails to take the offsetting nature of sentiment features into consideration will not accurately portray sentiment.

Fig. 1 shows a sample sentiment evaluation by Lexalytics, an NLP based Twitter sentiment service, which uses a −1.00 to +1.00 sentiment scale:

Lexalytics correctly identifies the object of the sentence, but the negative scores for the words "sick", "fever" and "cure" result in a negative score for the sentence as a whole, which ultimately fails to capture the author's intent. Lexalytics also misses one of the most common positive sentiment features in the Twitter-sphere,

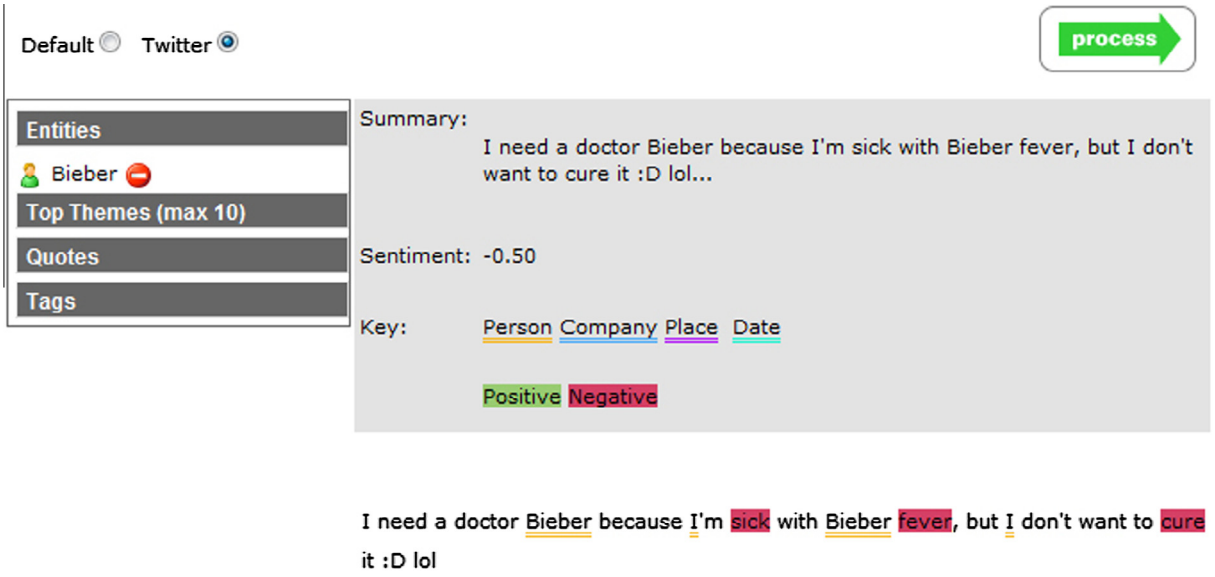

Fig. 1. Lexalytics sample sentiment evaluation results.

the emoticon (e.g. ":D"). In this example a strongly positive message is coded erroneously as negative by the engine.

The lack of context associated with a unigram based approach is resolved to some degree by moving to *n*-gram analysis. Practitioners employing an *n*-gram approach to sentiment analysis have found greater accuracy because the *n*-gram structure stores a greater degree of context and language complexity than a single word. At the 2011 Twitter DevNest conference, Jeffrey Kinsey, Co-Founder at DataMinr, reported that "the larger the *n*-gram the more accurate it is in determining sentiment". For example, if the tweet had been scored by comparing the word pairs found in the tweet to a set of known bi-grams that included the strongly positively bi-gram "Bieber fever", the scoring process would have assigned a strongly positive weight to the occurrence of "Bieber fever" in the tweet (instead of the negative score assigned to the "fever" singleton) and as a result the entire sentence would be classified as having positive sentiment. If a 5 member *n*-gram were to be used (e.g. a reference set that included the *n*-gram "I'm sick with Bieber fever") the analysis would yield an even more accurate determination of sentiment, however the larger the *n*-gram the fewer the number of sentence matches. Therefore, when applying *n*-gram analysis in a relatively short Twitter messages, the size of n in *n*-gram needs to be small. Kinsey reports that for his DataMinr service the optimal *n*-gram length is 3 (e.g. tri-gram). We also use $n = 3$ in our study.

4.6. Twitter-specific lexicon set

We offer a feature engineering method for Twitter feature selection and reduction. Most existing Twitter sentiment analysis approaches use a very large set of features. Applying a large feature set results in (a) a very sparse feature-tweet input matrix, and (b) less significant features contribute to the model noise level. These factors often reduce model accuracies in machine learning approach to sentiment analysis. We, therefore, offer a feature engineering approach that (a) reduces the feature set, (b) creates a denser feature-tweet input matrix, and (c) ensures that no significant feature is excluded from the final feature set. Our approach adheres to the feature engineering guidelines suggested by Mitchell (2005), who suggests that to maximize the predictive ability of a reduced set of features; one should consider the following six attributes: Information loss, Bias, Noise, Collision of negative and positive semantic vectors, Differences in Scale, and Overfitting.

We manually derived the most impactful terms from a large body of Twitter messages concerning a single subject (10^7 tweets related to Justin Bieber). We find there are a distinct set of features that have a greater sentiment value (i.e. "love", "hate") than others (i.e. "if", "but"). We also find that sentiment terms are not well distributed throughout the dataset. Strongly negative terms are found approximately 1/8th as often as strongly positive terms. Any feature reduction approach that requires well distributed data will struggle to maximize the predictive ability of a set of features drawn from social media messages.

We have developed a Twitter-specific approach for feature selection for our analysis. The process uses linguistic analysis to determine the *n*-grams of interest, and validates the discovered *n*-grams using frequency and affinity metrics (defined below). As stated earlier we used the Twitter API to collect our dataset. The dataset included all tweets between May 6th, 2012, 7am and June 8th, 7am, 2012 (that mentioned @justinbieber). A total of 10,345,184 tweets were gathered. We then divided the tweets into groups based on the last two digits of the tweet ID. This was done in order to select random groups of tweets that spanned the entire investigation period and to limit the influence of "spam" tweets. A "spammer" will send the same tweet as many as 100 times in a row, thus an algorithm that only takes one row out of every 100 sequential tweets will reduce the impact of a "spammer" (to that of a normal follower).

Ten groups were selected as candidate groups: candidate groups ranged in size from 1364 to 57,319 tweets. Tweet distribution was then evaluated across the 10 groups (based on time of day and day of month). The goal was to find a candidate group that most closely resembled the distribution of tweets in the greater dataset. A group of tweets with IDs ending in 46 was selected (e.g. Tweet ID 66978297401196546 which reads,"@justinbieber if u follow me this will b the best day of my life!!!" was included in the dataset). The model group was initially composed of 12,505 tweets. Re-tweets of Justin's messages to his followers were first removed from this set so that the sentiment Justin directly expressed (in his tweets) would not be reflected back onto him. This resulted in a model group of 9749 tweets. To provide sufficient instances to train our classification models, we aimed to score approximately 3000 tweets of the 9749 by our human evaluators.

Next, in order to train the sentiment engine, training and testing datasets are generated. The sets include representative tweets from each class, while accounting for unbalanced nature of tweet sentiment classes. Finally, these tweets are manually scored and classified. Three Information Systems graduate student evaluators independently scored 3000 tweets for sentiment on a 5 step positive to negative scale. Scale steps were "Strongly Positive", "Mildly positive", "Neutral", "Mildly Negative", and "Strongly Negative". Sentiment evaluations that did not agree between the evaluators were discarded, resulting in 2618 tweets incorporated into the final dataset. Evaluators also marked the most meaningful bi-grams and tri-grams in each tweet. This step was performed in order to build the lexicon that would become the basis of the feature set.

A statistically significant number of tweets in each sentiment category were required in order to fully evaluate the performance of the DAN2 and SVM models. However, the "Strongly Positive" bucket was found to so thoroughly dominate the 2618 tweets that additional refinement in the final dataset was required to account for the unbalanced categories. We required each category to have at minimum 500 representative instances in the final datasets. To reach this threshold, a supplementary set of 300 negative tweets were also included in the final dataset. To acquire these additional negative instances, we extracted another set of tweets from the 9749 model group not already included in the dataset to be coded by our team of human evaluators, and incorporated 300 negative tweets with agreement among the evaluators into the final datasets. Therefore the final, refined model was composed of 2918 tweets with sentiment (2618 random tweets with sentiment across all five categories +300 negative tweets filtered from another set). This group was used as the basis for our analysis.

We next evaluated the bi-grams and tri-grams that were identified as most meaningful, in the previous stage, and decomposed them into unigrams and at this point unigram term frequency was calculated across the refined dataset. For example, Tweet ID 66978297401196546 was determined to include the meaningful tri-gram "best day ever", which was then decomposed into "best", "day" and "ever". After application of term frequency across the set of unigrams, the most frequently occurring terms (terms with frequency $\geqslant 0.033\%$) were selected as features (standard stop words, pronouns and quantity determiners were excluded). The goal of term frequency evaluation was to arrive at a reduced lexicon that could be generically applied to Twitter messages by future investigators. This approach is in sharp contrast to Go et al. (2009) in which no consideration was given to the ability to supervise the lexicon that was employed.

Since the term frequency parameter (0.033%) is experimentally defined, and to ensure that the resulting reduced feature set has general applicability, additional terms were added to feature set (in order of decreasing term frequency) until nearly all tweets in the full 10,345,184 row corpus contained one or more of the featured terms. The specific goal was 97% of all tweets sent to Justin Bieber during the period in question had to contain one or more of the features in the feature set. This resulted in a feature set containing approximately 700 of the most frequently occurring terms and emoticons that were required in order to meet the coverage metric.

At this point the feature set had not been cleansed in any way, stemming had not occurred, and the precision of the original bi-grams and tri-grams had been traded for the unigrams' higher term frequency. We next developed a feature boundary for each term. To determine the appropriate boundary, the affinity of each term to each sentiment class was examined. Any term that had a strong affinity for one side of the sentiment scale (e.g. frequently showing up in positive sentences and rarely showing up in negative sentences) was kept. Any term that did not clearly indicate sentiment was either set aside for later *n*-gram engineering or dropped altogether. For the terms that remained, synonyms were found and added to the feature definition. The feature boundary work resulted in features that were both highly explanatory and combinable with other features of similar sentiment affinity. At this point the feature set fell below our target 97% coverage goal due to the removal of ambiguous terms.

We then examined contradictory terms and did not allow them to co-exist in the same feature, as this would severely impact the feature's explanatory ability. A good example of this problem is the term "hate". A stemmed hate (i.e. "hate%") picks up all occurrences of "hate" and "haters". In the Justin Bieber corpus, the term "haters" is almost universally used to describe people that are seen in a negative light because they do not like Justin (i.e. they hate Justin, but the author loves Justin, so the author describes the group with the pejorative term "haters"). As a result messages regarding "haters" predominantly hold positive sentiment toward Justin. Because of this contradiction the "hate" feature includes a "hate" search that is not stemmed (i.e. "hate"), and a different, opposing feature called "haters" searches for the stemmed term "haters" (i.e. "haters%").

Another issue that arose once the selected bi-grams and tri-grams were abandoned involved negations. Terms that are negated are difficult to handle with pure unigram analysis. At the minimum the sentence "I'm not happy" requires a "not happy" feature to be properly scored (as either a bi-gram or two related unigrams). Go et al. (2009) attempted to deal with negation using bi-grams to no avail. We successfully included negation in our model by employing frequently occurring negations (in bi-gram form) and frequently occurring negative terms in the singular (in unigram form). We created specific features for the most common sentiment negations ("not happy", "not good", "not great", "not impressed", "not nice", "not like", "not disappoint", "not enjoy", "can't wait", "can't stand", "don't want", "no reason" and "no chance") as well as features for each not variant itself ("not", "can't", "cannot", "isn't", "doesn't", "don't", and "won't"). This approach follows the method used by Pak and Paroubek (2010) who proposed, "A negation (such as "no" and "not") is attached to a word which precedes it or follows it. For example, a sentence "I do not like fish" will form two bi-grams: "I do + not", "do + not like", "not + like fish." Such a procedure allows improving the accuracy of the classification since the negation plays a special role in an opinion and sentiment expression."

Once the original bi-grams and tri-grams were decomposed, and a set of unigrams were selected based on term frequency, the most frequently occurring bi-grams and tri-grams could then be "safely" introduced back into the model (without risk of over fitting) using the affinity method, championed by Kajanan, Shafeeq Bin Mohd Shariff, Datta, Dutta, and Paul (2011). They define the *Affinity* of a word phrase *P* as

$$\textit{Affinity}\ (P) = \frac{f(P)}{min_{\forall wi \epsilon P}(fWi)}$$

where $f(P)$ is the frequency of phrase P in a corpus, and $\min(f(w_i))$ is the minimum frequency across the words in a phrase P in the corpus. The affinity approach adds back bi-grams and tri-grams that have "higher collocation frequencies relative to individual occurrence frequencies of the constituent unigrams" ensuring that the additions are driven by *n*-gram frequency and meaningfulness. To further manage bias in the feature set, bi-grams and tri-grams were only added to features that were still lacking in precision after the initial refinement. For example, the stemmed "hope%" search term was not as precise as the following set of *n*-grams {"%hopeful%", "%I hope u%", "%I hope you%", "%I hope he%", "% hoping%", "%do not lose hope%", "%don't lose hope%", "%don't lose hope%"} and was replaced with the *n*-gram set.

Meaningful bi-grams and tri-grams were added to the feature set in order to once again regain its 97% coverage rate. This resulted in a total feature set of 755 *n*-grams (*n* = 1, 2, 3). After the feature engineering phase, we then combined like features to arrive at 187 feature "groups" that each had a high degree of affinity for either positive or negative sentiment. Feature groups were formed around groups of *n*-grams with similar meaning. For example, the POS_LOVE feature group is composed of the *n*-grams "love", "lovin" and "luv". The POS_THANK feature group is composed of the *n*-grams "thank", "thx" and "thks". The POS_HI feature group includes 9 variants of "hi". In a limited set of cases the grouping includes *n*-grams that have some minor degree of difference in terms of meaning, but were found to be used interchangeably on the Twitter service. For example the POS_AMAZIN feature group is composed of "amazin", "amaze" and "brilliant". At this juncture the reduced feature set was able to characterize 97.3% of the 10,345,184 messages in the greater Twitter corpus (i.e. one or more of the 187 feature "groups" appeared in 97.3% of the tweets). This indicates that no explanatory power was lost in the reduction to the 187 features (as the groups were simply logical partitioning of the 755 features). This produced a significantly smaller and denser input matrix for our analysis. Features were considered present or not present in the matrix (0 or 1). This process is an "equivalence partitioning" approach in which a representative feature is used as a proxy for a set of features with similar sentiment effect. Multiple presences of such features in a message are still presented with a single attribute of one in the input matrix. We also note that repetition of a single feature within a message did not add explanatory power (e.g. one happy face emoticon was grammatically identical to many happy face emoticons), thus repetition was not considered useful and was ignored. For example, the positive emoticon feature ("happy_emoticon") was based on the 46 emoticons that were most frequently found in the positively-scored tweets. A tweet with the following text "@justinbieber ☺☺☺☺☺☺☺" resulted in the same feature score as the tweet "@justinbieber ☺" (both resulted in the feature happy_emoticon = 1).

Our approach minimizes information loss and noise by employing a model with a supervised feature set (187 unigrams, bi-grams and tri-grams).

4.7. Feature selection walk through

Our feature selection process chooses high frequency terms that have a clear affinity for either positive or negative sentiment (e.g. "love", "hate") and discards terms that are ambiguous (e.g. "from", "again"). Further refinement of the feature set combines like terms, adds synonyms, and solves for the cases of contradiction and negation. Finally, related bi-grams and tri-grams are then found using

Table 2
Final Twitter feature list.

POS_ADORE	POS_AMAZIN	POS_AMEN	POS_AWARD
POS_APPRECIATE	POS_AWESOME	POS_BABY	POS_BEAUTIFUL
POS_BELIEF	POS_BEST	POS_BETTER	POS_BIGGEST
POS_BLESS	POS_BRIGHT	POS_CANT_WAIT	POS_CHECK_IT_OUT
POS_CONGRAT	POS_COURAGE	POS_CRUSH	POS_CUTE
POS_DREAM	POS_DYING	POS_EXCITE	POS_FEEL
POS_FEVER	POS_FAVORITE	POS_FRIEND	POS_FUNNY
POS_GREET	POS_HAHA	POS_HATERS	POS_HEART
POS_HELL_YES	POS_HELLO	POS_HERO	POS_HI
POS_HILARIOUS	POS_HONEST	POS_HOPE	POS_HOT
POS_INSPIRE	POS_KILL	POS_LAUGH	POS_LIKE
POS_LOL	POS_LOVE	POS_LUCK	POS_MAKE_MY
POS_MISS	POS_MISSING	POS_MYSELF	POS_NO_MATTER_WHAT
POS_NOT_DISAPPOINT	POS_NEVERSAYNEVER	POS_PARTY	POS_PEACE
POS_PERFECT	POS_POSITIVE	POS_PRAY	POS_PRETTY
POS_PROPS	POS_PROUD	POS_RESPECT	POS_RUMOR
POS_SEX	POS_SMILE	POS_SORRY	POS_SPECIAL
POS_SUCCESS	POS_SUPER	POS_SUPPORT	POS_SURE
POS_SWAG	POS_TALENT	POS_THANK	POS_TREND
POS_TRUTH	POS_WAITING	POS_WANT	POS_WATCH
POS_WELCOME	POS_WINNER	POS_WISH	POS_WITHOUT_YOU
POS_WONDERFUL	POS_WOW	POS_YES	POS_OTHER
POS_OTHER2	POS_NOTTEST_ENJOY	POS_NOTTEST_GOOD	POS_NOTTEST_GREAT
POS_NOTTEST_HAPPY	POS_NOTTEST_IMPRESSED	POS_NOTTEST_NICE	NEG_NOT_GOOD
NEG_NOT_HAPPY	NEG_NOT_LIKE	NEG_ANNOY	NEG_BITCH
NEG_CANT_STAND	NEG_CONFUSE	NEG_COMPLAIN	NEG_CRAP
NEG_CRY	NEG_DAMMIT	NEG_DISAPPOINT	NEG_DISLIKE
NEG_FAIL	NEG_FAKE	NEG_FUCK	NEG_GONE
NEG_GRR	NEG_HORRIBLE	NEG_HURT	NEG_I_HATE
NEG_I_DONT_WANT	NEG_IDIOT	NEG_IGNORE	NEG_IMPOSSIBLE
NEG_JUST_SAYING	NEG_LAME	NEG_LET_DOWN	NEG_LIAR
NEG_LOSE	NEG_NEGATIVE	NEG_NO_REASON	NEG_NO_CHANCE
NEG_NOT_NICE	NEG_RAIN	NEG_RUDE	NEG_RIDICULOUS
NEG_SAD	NEG_SCREW	NEG_SELFISH	NEG_SHAME
NEG_SHIT	NEG_SLUT	NEG_STUPID	NEG_SUCK
NEG_TERRIBLE	NEG_TOO_BAD	NEG_TRANSACTION_TERMS	NEG_UGLY
NEG_USELESS	NEG_WEIRD	NEG_WHY_WONT	NEG_WORST
NEG_WTF	NEG_YOU_WONT	NEG_OTHER	NEG_OTHER2
NEG_NOTTEST_ANGRY	NEG_NOTTEST_UPSET	NEG_NOTTEST_PISSED	NEG_NOTTEST_MAD
MIXED_BUT	MIXED_DEPRESSED	MIXED_DEATH	MIXED_NOT_FEELING_WELL
MIXED_IF	MIXED_DAMN	MIXED_CRAZY	MIXED_JEALOUS
MIXED_OMG	NOTTEST_NOT	NOTTEST_ISNT	NOTTEST_ISN_T
NOTTEST_DOESNT	NOTTEST_DOESN_T	NOTTEST_DONT	NOTTEST_DON_T
NOTTEST_WONT	NOTTEST_WON_T	NOTTEST_CANT	NOTTEST_CAN_T
NO	QUESTION	PLEASE	EXCLAIM
REALLY	FOLLOW	MESSAGING	HAPPY_EMOTICON
SAD_EMOTICON	isHTTP	isSPANISH	

Kajanan's *Affinity* algorithm (meaningful *n*-grams that exhibited higher collocation frequencies relative to individual occurrence frequencies of the constituent unigrams). These feature engineering steps have resulted in 187 features listed in Table 2. The prefix of "POS_" or "NEG_" indicates the sentiment affinity of the feature. The prefix of "NOTTEST" indicates a contradiction (these terms are used as modifiers of sentiment). The prefix of "MIXED" indicates a feature that does not have strong affinity in either direction (no clear boundary could be arrived at for this feature but it is instrumental in describing sentiment, often negating a sentence or adding a condition to it). The prefix of "is" denotes information about the Tweet itself (e.g. it is written in Spanish, or it does include an HTTP link). And finally there are nine features that occur with such high frequency they are not assigned a sentiment prefix (though they are still considered to contain sentiment). The frequently occurring features include positive emoticons, negative emoticons, "please" and "follow".

4.7.1. Re-use of the lexicon

One of the goals of our research was to arrive at a lexicon that would enable ongoing investigation of sentiment on the Twitter service, regardless of message domain. In the lexicon set we developed only six features were specific to Justin Beiber. These features (POS_FEVER, POS_NEVERSAYNEVER, POS_OTHER, POS_OTHER2, NEG_OTHER, and NEG_OTHER2) contained terms that we would not expect to find in other brands' Twitter feeds. The remaining 181 features are highly descriptive of sentiment throughout the Twitter-verse; however it will be left to future investigations to quantify the usefulness of this lexicon.

4.8. Customization specific to the Twitter service

The messages on the Twitter service often involve functional elements of the Twitter service itself. A successful *n*-gram model, then, must include proper handling for terms that describe Twitter-specific behaviors:

Re-tweets – The act of repeating an author's message to others

Follow requests – The request for an author to add a user to a privileged list (i.e. a list of users the author follows)

In the analysis of the Justin Bieber's Twitter feed we found that 30.31% of all messages pertained to follow requests, thus the explanatory value of the term "follow" cannot be ignored. We also found that 18.93% of all messages that mentioned @JustinBieber were re-tweets (of an original message authored by Justin). The impact of Justin's own sentiment in re-tweet message must be handled appropriately as well. Identifying these elements in the model is surprisingly quite simple. To identify re-tweets the model simply has to look for "RT", the acronym that appears before a re-tweeted sentence. To identify follow requests the model simply must search for occurrences of "follow" and the misspelled variant "folllow".

Analysis of the messages revealed that re-tweets of Justin Bieber's words did not have significant new information (apart from the original message). What little information was added to the original message had a strong positive bias as well. It is a trivial act to predict that these messages will be positive, and as such no further investigation of these messages was required. As previously mentioned, the corpus that was at the heart of our investigation subsequently had re-tweets filtered out.

4.9. Inclusion of emoticons

The model includes the character combinations that are used to express sentiment on the service, commonly referred to as emoticons. While some researchers have dismissed emoticons as noisy and not containing adequate information, our analysis demonstrated that the model benefits from the information held in the emoticon data.

Table 3
Happy and sad emoticons.

Happy emoticons	Sad emoticons						
:P	:D	:d	:p	:(	;(	:'(	;'(
;P	;D	;d	;p	=(	={	):	);
:-)	;-)	:~)	;~)	)':	)';	)=	}=
:<)	:>)	;>)	=)	;-{{	;-{	:-{{	:-{
=}	:)	(:	;)	:-(	;-(	:~-(	;~-(
(;	:}	{:	;}	:,(	:'{		
{;	:]	[:	;]				
[;	:')	;')	:-3				
;-3	:-x	;-x	:-X				
;-X	:-}	;-}	:-]				
;-]	:-.)	♥	☺				
^_^	^_^						

The commonly employed happy and sad emoticons are presented in Table 3:

Our feature set includes these commonly used emoticons. Each group of emoticons is represented by a single, distinct feature in the feature set. Emoticons are represented in the model by the features HAPPY_EMOTICON and SAD_EMOTICON.

We offer a note on the neutral category. Using our strict interpretation of neutral sentiment ("unclear how the author feels about the brand") our team of evaluators found that only 10.2% of Twitter messages referring to Justin Bieber were neutral. Since our use case focused on identifying the differences between strongly positive, mildly positive, mildly negative and strongly negative messages we decided to limit the DAN2 and SVM models to these four sentiment classifications. Tweets that DAN2 or SVM could not fit into one of the four sentiment classifications were considered neutral.

Once the tweets are fully preprocessed and feature engineering techniques have been applied, the tweets are randomly divided into training and testing datasets and the classification process can begin. The classification strategies used for analysis are binary and multi-class categorization. In binary classification only two categories are present: either the tweet belongs to the class or it does not. This strategy is called "one vs. all" method and is used in this study. Most multi-class classification strategies are multiple iterations of binary classification.

Finally, we note that the feature engineering approach introduced in this section has produced a reduced (general) Twitter-specific and some brand (Bieber) specific lexicon. The reduction of features is substantial and will be shown to be still very effective. The reduced lexicon set offers the following benefits:

1. The specialized set simplifies the modeling task by reducing complexity of the models (model size reduction).
2. The reduced lexicon set still produces excellent coverage with very accurate results.
3. Although, we only report results based on one dataset, the effectiveness measures (recall and accuracy values) are better than others reported in the literature for similar datasets/analysis.

To show the effectiveness of the reduced lexicon set, in Section 5.4 we use the standard lexicon set traditionally employed by other researchers with SVM for our dataset and compare its effectiveness against the reduced lexicon set. Our results show that the reduced lexicon set is more effective while offering the benefits listed above. This conclusion is consistent with studies in other

fields such as financial and accounting analysis that also relies on a reduced lexicon set (Das & Chen, 2007; Loughran & McDonald, 2011).

4.10. Sentiment scale

We developed a scale that is both precise and still easy to supervise. The sentiment scale has sufficient steps such that a brand marketer can identify extremely positive and extremely negative statements regarding his or her product. It is expected that the extremely positive messages would be promoted by the brand, and the extremely negative messages would be referred to customer service to resolve. In light of this expected use case a broad neutral category is therefore not deemed desirable.

In our scale we represent sentiment with a set of labels. We chose to use labels that traverse a linear scale (negative to positive) as opposed to labels that categorizes tweets into multiple moods. We find that a "positive to negative" scale is less hampered by interpretation bias.

We employ a scale that is more precise than the three step scale and easier to supervise than Lexalytics' 200 step scale. Our scale uses five steps [−2, −1, 0, 1, 2] to categorize sentiment. The definition of each step is as follows:

2 = author clearly loves the brand (Strongly Positive)
1 = author likes the brand (Mildly Positive)
0 = unclear how author feels about the brand (Neutral)
−1 = author dislikes the brand (Mildly Negative)
−2 = author clearly hates the brand (Strongly Negative)

The simplicity of this scale was of great benefit during the manual scoring process. We find that the easier it is to score a sentence the less chance a sentence will receive different scores by different human evaluators.

4.11. Training and testing datasets and accuracy metrics

We generated a manually classified dataset for this analysis. The dataset was randomly selected from the Twitter corpus. A total of 3440 tweets were manually classified by the three graduate students. We only used tweets that were similarly classified by all three. This resulted 2918 manually scored tweets. We next randomly partitioned this dataset into training (51%) and testing (49%) datasets. The training dataset was used to train the models and the testing dataset was kept out for independent testing of the trained model. As stated earlier, we used a "one vs. all" approach for classification. Therefore, we developed four separate models: one each for strongly positive, mildly positive, strongly negative, and mildly negative classes. The training dataset was used to calculate the starting point and training of each model. The same training and testing datasets were used for both the DAN2 and SVM models

The training dataset is used to develop the four different models for Twitter sentiment analysis. To gauge the effectiveness of the trained models the recall and accuracy for the prediction of each sentiment value was measured. The overall accuracy (across all sentiment buckets) was measured as well. The recall statistic for a given sentiment value X is defined as:

(Number of tweets correctly predicted to have given sentiment value X)/(Total number of tweets with given sentiment value X)

While the accuracy statistics for a given sentiment value X is defined as:

(Number of tweets correctly predicted for X + number of tweets correctly predicted to not be in X)/(Total number of tweets)

We note that precision is not a valid metric to apply to this approach, as 100% of rows (Tweets) are categorized (every single row falls into a sentiment value).

Finally, we also note that any model training should avoid overfitting. Both DAN2 and SVM use internal metrics to avoid overfitting.

5. Automated, supervised sentiment analysis

As stated earlier, we consider Twitter sentiment analysis as a text classification problem. We use DAN2 and SVM as two methods for this analysis. These methods are supervised machine learning approaches that require a training dataset for their learning stage. Once each model is trained, they can be used to automatically provide sentiment associated with previously unseen input (tweets).

Once the feature set is defined, in order to assess a tweet's or a corpus's sentiment, a functional form and a set of associated weights needs to be computed. Determining the functional form and such a set of weights is challenging. Generally, we define the Twitter sentiment as:

$$TS_i = f(W_j * \text{Feature}_j) \quad \text{where } i = \text{Tweet}i, \text{ and } j = \text{feature},j = 1, \ldots, 187$$

In previous sections we introduced a feature engineering approach for term selection and a five point sentiment scale for tweets. To use DAN2 and SVM we needed to create training and testing datasets. Creation of these datasets requires human classification of tweets. We developed a tool that allowed three graduate students to manually classify tweets into one of the five classes by reading a tweet and using the guidelines and the tool workflow to classify it. The tool used by the graduate students for classification presented a single tweet at a time for review. The tweet was displayed in two different sections of the screen, at the same time, in two modes: (1) without emoticons and (2) with emoticons. This allowed the students to review the language of the tweet separate from the tweet's use of emoticons. This was done in order to prevent a quick, automatic classification on the part of the student, based solely on recognition of an emoticon. The student then had to answer 2 questions regarding the tweet ("is it English?" and "Does it contain sentiment?") before the sentiment scale was presented for coding. Again, these workflow steps were introduced in order to force the classifier to seriously consider the content of the tweet and prevent a quick classification.

An identical set of 3440 tweets were scored by each student. The guidelines for categorization were as follows:

- Strongly Positive – the author loves Justin Bieber or his music.
- Mildly Positive – the author likes Justin Beiber or his music.
- Neutral – the author has no opinion of Justin Bieber.
- Mildly Negative – the author dislikes Justin Bieber or his music.
- Strongly Negative – the author hates Justin Bieber or his music.

This experiment produced 2918 manually classified tweets for which the classifiers were in agreement. There was classification disagreement on 522 tweets (15.2%).

We note that the human classification of tweets did not rely on nor had knowledge of the 187 features introduced in the feature engineering section.

5.1. DAN2 for Twitter sentiment analysis

We have developed a neural network model, DAN2, (***A Dynamic Architecture for Artificial Neural Networks***), which employs a different architecture than the traditional neural network (FFBP) models. The general philosophy of the DAN2 model is based upon the

principle of learning and accumulating knowledge at each layer, propagating and adjusting this knowledge forward to the next layer, and repeating these steps until the desired network performance criteria are reached. Fig. 2 presents the overall DAN2 architecture.

As in classical neural networks, the DAN2 architecture is composed of an input layer, hidden layers and an output layer. The input layer accepts external data to the model. In DAN2, unlike classical neural nets, the number of hidden layers is not fixed a priori. They are sequentially and dynamically generated until a level of performance accuracy is reached. Additionally, the proposed approach uses a fixed number of hidden nodes (four) in each hidden layer. This structure is not arbitrary, but justified by the estimation approach. At each hidden layer, the network is trained using all observations in the training set simultaneously, so as to minimize a stated training accuracy measure such as mean squared error (MSE) value or other accuracy measures. As shown in Fig. 2, each hidden layer is composed of four nodes. The first node is the bias or constant (e.g. 1) input node, referred to as the C node. The second node is a function that encapsulates the "Current Accumulated Knowledge Element" (CAKE node) during the previous training step. The third and fourth nodes represent the current residual (remaining) nonlinear component of the process via a transfer function of a weighted and normalized sum of the input variables. Such nodes represent the "Current Residual Nonlinear Element" (CURNOLE nodes). In Fig. 2, the "*I*" node represents the input, the "*C*" nodes are the constant nodes, the "G_k" and "H_k" nodes represent CURNOLE nodes, and the "F_k" nodes are the CAKE nodes. The final CAKE node represents the dependent variable or the output.

At each layer, the previous four nodes (C, G_k, H_k, and F_{k-1}) are used as the input to produce the next output value (F_k). The parameters on the arcs leading to the output nodes, (a_k, b_k, c_k, d_k), represent the weights of each input in the computation of the output for the next layer. The parameter connecting the CURNOLE nodes, μ_k, is used as part of the argument for the CURNOLE nodes and reflects the relative contribution of each input vector to the final output values at each layer. A detailed description of the architecture and its properties are fully presented in (Ghiassi & Saidane, 2005).

The training process begins with a special layer where the CAKE node captures the linear component of the input data. Thus, its input (content) is a linear combination (weighted sum) of the input variables and a constant input node. These weights are easily obtainable through classical linear regression. If the desired level of accuracy is reached, we can conclude that the relationship is linear and the training process stops. This step is used as the starting point. For classification problems this step can be replaced with alternative methods described in Section 5.1.1. For nonlinear relations additional hidden layers are required. At each subsequent layer the input to the CAKE node is a weighted sum (linear combination) of the previous layer's CAKE, CURNOLE, and C nodes. Throughout training, the CAKE nodes carry an adequate portion of learning achieved in previous layers forward. This process ensures that the performance or knowledge gained so far is adjusted and improved but not lost. This property of DAN2 introduces knowledge memorization to the model. Ghiassi and Saidane (Ghiassi & Saidane, 2005) show that the DAN2 algorithm ensures that during network training, the residual error is reduced in every iteration and the accumulated knowledge is monotonically increased.

The training process defines creation of partitions among classes that could include linear and nonlinear components. The linear component of the input data is captured in the first CAKE node using ordinary least squares (OLS) or other simple and easy to compute approaches. The algorithm next transforms the input dataset to model the nonlinearity of the process in subsequent iterations. DAN2 uses a vector projection approach to perform data transformation. The transformation process defines a reference vector $R = \{r_j; j = 1, 2, \ldots m\}$, where m represents the number of attributes of the observation records, and projects each observation record onto this vector to normalize the data as discussed in (Ghiassi & Saidane, 2005). This normalization defines an angle, α_i, between record i and the reference vector R. DAN2 uses the set of α_i's to train the network, and updates their values in every iteration. In Ghiassi and Saidane (Ghiassi & Saidane, 2005), the authors show that this normalization can be represented by the trigonometric function Cosine $(\mu_k\alpha_i + \theta_k)$. In every hidden layer k of the architecture we vary $(\mu_k\alpha_i + \theta_k)$ and measure the impact of this change on the output value. The modification of the angle $(\mu_k\alpha_i + \theta_k)$ is equivalent to rotating μ_k and shifting θ_k, the reference vector, thus changing the impact of the projected input vectors and their contribution to the output for that iteration. The Cosine $(\mu_k\alpha_i + \theta_k)$ uses two (nonlinear) parameters, μ_k and θ_k. The use of the latter can be avoided through the expansion of the cosine function in the form: A Cosine $(\mu_k\alpha_i) + B$ Sine $(\mu_k\alpha_i)$. We use this functional form as the transfer function in our model. The two CURNOLE nodes in Fig. 2 represent this formulation. At any given hidden layer k, if the Cosine $(\mu_k\alpha_i + \theta_k)$ terms captured in previous layers do not adequately express the nonlinear behavior of the process, a new layer with an additional set of nodes is automatically generated, including a new Cosine $(\mu_k\alpha_i + \theta_k)$ term. This process is analogous to how the Fourier series adds new terms to improve function approximation. Therefore, the number of layers in the DAN2 architecture is dynamically defined and depends upon the complexity of the underlying process and the desired level of accuracy. Thus, the output of this model is

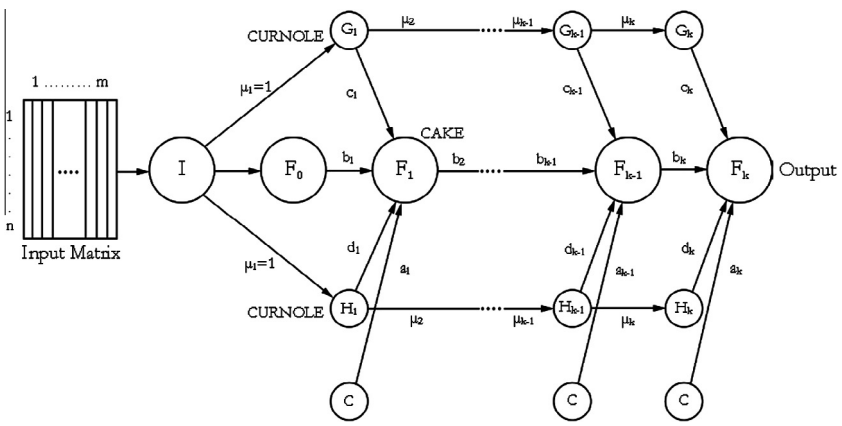

Fig. 2. The DAN2 network architecture.

represented by the linear combination of the constant, CAKE, and CURNOLE nodes. Eq. (1) represents the functional form of this relationship at iteration (layer) k

$$F_k(X_i) = a_k + b_k F_{k-1}(X_i) + C_k G_k(X_i) + d_k H_k(X_i) \quad (1)$$

where X_i represents the n independent input records, $F_k(X_i)$ represents the output value at layer k, $G_k(X_i)$ = Cosine ($\mu_k \alpha_i$), and $H_k(X_i)$ = -Sine ($\mu_k \alpha_i$) represent the transferred nonlinear components, and a_k, b_k, c_k, d_k, and μ_k are parameter values at iteration k. The training process initially captures the linear component by using OLS or other simple and easy to compute approaches. If the desired level of accuracy is reached, the training terminates. Otherwise, the model generates additional layers to capture the nonlinear component of the process by minimizing a measure of total error as represented by $SSE_k = \Sigma_i[F_k(X_i) - \hat{F}(X_i)]^2$. Substituting $F_k(X_i)$ from Eq. (1) results in:

$$SSE_k = \sum_i \left[a_k + b_k F_{k-1}()X_i + C_k \cos(\mu_k \alpha_i) + d_k \sin(\mu_k \alpha_i) - \hat{F}(X_i)\right]^2 \quad (2)$$

where $\hat{F}(X_i)$ are the observed output values. Minimizing Eq. (2) requires the estimation of five parameters. This formulation is linear in the parameter set A_k, where $A_k = \{a_k, b_k, c_k, d_k\}$ and nonlinear in parameter μ_k. In Ghiassi and Saidane (Ghiassi & Saidane, 2005), they present several nonlinear optimization strategies to estimate the nonlinear parameter μ_k. We also show that following this approach, at each layer the knowledge gained is monotonically increased, total error is reduced, and the network training improves.

DAN2's algorithm is highly scalable. The algorithm is composed of a series of layers that are automatically and dynamically generated. At each layer the observation vectors are projected into the reference vector to create the angle α_i as discussed earlier. The training and optimization at each layer only uses the transformed data to estimate the five parameters of Eq. (2). Therefore, regardless of the original size of the dataset, DAN2 only needs to compute the set of four linear parameters, $A_k = \{a_k, b_k, c_k, d_k\}$, and one nonlinear parameter μ_k at each iteration. For instance, for text classification problems with large feature sets, once the starting point is computed, each observation record, as represented by its many features, is projected onto the reference vector and its angle, α_i, is computed. The training process only uses the α_i values and only needs to compute the five parameters. Therefore, the original problem with a large feature set is converted to a 5-parameter model at each layer; thus scaling the problem size and demonstrating the scalability of the model. The scalability of DAN2 is a distinguishing strength of the approach from traditional artificial neural networks.

In (Ghiassi, Saidane, & Zimbra, 2005; Ghiassi & Burnley, 2010), the authors compare DAN2 with traditional FFBP and recurrent neural network (RNN) models. The comparison spans both theoretical and computational perspectives using several benchmark datasets from the literature. Performance of DAN2 against these models as well as non-neural network alternatives is also presented. Their studies show that DAN2 outperforms all other alternatives and produces more accurate training and testing results in every case.

5.1.1. Starting point for DAN2

Creating a good starting point in any nonlinear optimization problem is always challenging. However, good starting point values often improve nonlinear optimization convergence. For this analysis, we introduce a solution that is quick to implement and it is still better than MLR or a random assignment of classes to tweets. Our heuristic begins by further reducing the feature set from 187 to 17 by grouping features with identical sentiment weights into a single "auto-sentiment" group (Table 4).

Table 4
List of feature groups.

PLEASE
FOLLOW
AUTOSENTIMENT_POS1_TOTAL
AUTOSENTIMENT_POS2_TOTAL
AUTOSENTIMENT_POS3_TOTAL
AUTOSENTIMENT_NEG1_TOTAL
AUTOSENTIMENT_NEG2_TOTAL
AUTOSENTIMENT_NEG3_TOTAL
NEG_TRANSACTION_TERMS
MIXED_IF_lt
MIXED_DEPRESSED
MIXED_OMG
HAPPY_EMOTICON
SAD_EMOTICON
AUTOSENTIMENT_NOT_TOTAL
MESSAGING
isHTTP

Table 4 presents one feature with special meaning (isHTTP), and 16 groups. For example the feature group AUTOSENTIMENT_POS1_TOTAL holds the count of all mildly positive features; the group AUTOSENTIMENT_POS2_TOTAL holds the count of all strongly positive features, etc. We next represent the sentiment function as a linear function of the 17 features and define a simple set of weights of (1, 2, 4, 8, 10) for positive sentiments and (−1, −2, −4, −8, −10) for negative ones. These values are arbitrary and are only used to produce starting point values quickly. Positive features were assigned a value of .1 higher than their negative equivalent, so that offsetting terms would result in a mildly positive score. We add a bias value of 100 to the equation so positive sentiment tweets score above 100 and negative sentiment tweets score below 100. The resulting starting point equation is represented by Eq. (3).

$$\begin{aligned} \text{SENTIMENT\_SCORE} = {} & (100 + (\text{isHTTP} * (4.1)) \\ & + (\text{MESSAGING} * (-2)) + (\text{MIXED\_IF\_lt} \\ & * (1.1)) \\ & + \text{AUTOSENTIMENT\_NOT\_TOTAL} \\ & + (\text{NEG\_TRANSACTION\_TERMS} * (-4)) \\ & + (\text{PLEASE} * 4.1) + (\text{FOLLOW} * 2.1) \\ & + (\text{HAPPY\_EMOTICON} * 8.1) \\ & + (\text{SAD\_EMOTICON} * (-4)) \\ & + (\text{AUTOSENTIMENT\_POS1}_{\text{TOTAL}} * 4.1) \\ & + (\text{AUTOSENTIMENT\_POS2}_{\text{TOTAL}} * 8.1) \\ & + (\text{AUTOSENTIMENT\_POS3\_TOTAL} \\ & * 10.1) \\ & + (\text{AUTOSENTIMENT\_NEG1\_TOTAL} \\ & * 4) \\ & + (\text{AUTOSENTIMENT\_NEG2\_TOTAL} \\ & * 8) \\ & + (\text{AUTOSENTIMENT\_NEG3\_TOTAL} \\ & * 10) + (\text{MIXED\_DEPRESSED} * (-2)) \\ & + (\text{MIXED\_OMG} * (2.1))) \end{aligned} \quad (3)$$

This is a very simple equation for starting point computation and can be used to obtain a sentiment score for a tweet quickly. Once the score is calculated it is mapped to a sentiment class.

Positive (negative) scores are mapped into strong or mildly positive (negative) classes based on the sentiment score values. The resulting vector of class values is used as the starting point for DAN2 model.

5.1.2. DAN2 sentiment analysis results

This section presents the results of using the DAN2 classification approach for Twitter sentiment analysis. The input to DAN2 includes the vector representation of the tweets from the training dataset, their corresponding manual classification, and the starting point values. DAN2 uses the "one vs. all" classification approach for each given sentiment class. We, therefore, developed four separate models, one for each sentiment class (Tweets with sentiment that do not belong to any of the four classes are considered as neutral). The models are allowed to train on 1450 records in the case of the Strongly Negative category and 1500 records in the case of the other categories.

In order to ensure an even distribution between training and testing datasets, the Tweets are ordered by sentiment and then evenly divided between training and testing datasets (i.e. 1st row to train, 2nd row to test, etc.). This left an unbalanced set of Strongly Negative tweets at the tail end of the record set. In order to better balance the test set for the Strongly Negative model, 50 records are moved from the training dataset to the test dataset prior to building the Strongly Negative model. This resulted in a Strongly Negative training dataset with 50 fewer tweets and a Strongly Negative test set with 50 additional tweets.

The best DAN2 parameter values for each model are determined experimentally and the associated process is presented in Section 5.1.3. Models' performances are then evaluated using the recall and overall accuracy metrics. The distribution of the sentiment categories (positive and negative) in the training and testing datasets are shown in Table 5.

The unbalanced distribution of categories in the dataset has a material impact on the ability of all models; especially in identifying tweets in the less frequently occurring categories. Table 6 presents DAN2 results using the recall metric for both the training and testing dataset.

DAN2 exhibited relatively good recall values for all the classes, with the "Strongly Positive" and "Mildly Negative" classes showing the best results for the testing dataset (96.0% and 50.3%, respectively). Table 7 presents results of DAN2's performance using the accuracy metric. As indicated in Table 7, DAN2 performs well for all categories with the two "Negative" classes showing the best results for the testing dataset (89.9% and 95.1%, respectively).

Table 5
Distribution of categories in the full dataset.

Category	Category distribution (%)
Strongly positive	56.0
Mildly positive	26.7
Mildly negative	11.7
Strongly negative	5.6

Table 6
Recall by Category.

Category	DAN2 train (%)	DAN2 test (%)	SVM test (%)	SVM-of test (%)
Strongly positive	92.4	96.0†	94.1***	70.3
Mildly positive	25.8	11.5†††	0.0	0.0
Mildly negative	63.5	50.3†††	25.1***	0.0
Strongly negative	31.6	29.5†	19.3***	0.0

DAN2 vs. SVM
SVM vs. SVM-OF
† $p < 0.10$.
††† $p < 0.01$.
*** $p < 0.01$.

Table 7
Accuracy by category.

Category	DAN2 train (%)	DAN2 test (%)	SVM test (%)	SVM-of test (%)
Strongly positive	83.9	69.7†	71.3**	67.4
Mildly positive	85.0	66.7†††	63.9	64.0
Mildly negative	92.5	89.9†	88.3	87.3
Strongly negative	95.7	95.1	94.6***	93.7

DAN2 vs. SVM
SVM vs. SVM-OF
† $p < 0.10$.
††† $p < 0.01$.
** $p < 0.05$.
*** $p < 0.10$.

Table 8
Computation run-times for DAN2.

DAN2 computation	Duration (s)
Strongly positive all or nothing	8.4
Mildly positive all or nothing	6.4
Mildly negative all or nothing	4.8
Strongly negative all or nothing	4.9

5.1.3. DAN2 computational results

All machine learning algorithms, including DAN2 and SVM, require experimentation to find the best trained model. For example, for SVM, the value of the penalty parameter and the choice of the kernel function are selected experimentally (Joachims, 1999). Similarly, training of a DAN2 model requires experimentation. DAN2 initially defines a set of stopping conditions. These conditions include the maximum number of iterations and the target accuracy (the algorithm will stop at whichever condition is met first). At every iteration DAN2 solves a nonlinear optimization problem. In Ghiassi and Saidane (2005), the authors offer a number of alternatives to solve the intermediate nonlinear optimization problem. The implementation used in this research utilizes the grid search approach. In DAN2 a 2-dimensional search (interval and step sizes) is employed to find the best parameter value at each iteration. The algorithm begins with larger intervals and step sizes to determine promising search regions. When promising sectors are identified, subsequent more granular 2-D searches are performed on each sector to find the (local) optima. When a step produces inferior results, it is discarded and the process starts over with a different interval and step size. Once the desired level of training accuracy is reached, the search terminates. The algorithm uses internally set metrics to avoid overfitting as described in Ghiassi et al. (2005). We timed the computational effort required to train each category to assess DAN2's efficiency. Table 8 reports training times required for each of the four models. We note that although the size of the training set is an important variable in model training time, it is not the only significant factor. Clearly, the complexity of the problem, the underlying nonlinear model and the starting points, are also influential. As presented in Table 8, the largest category (Strongly Positive) takes the longest (8.4 s) to train. The shortest training times were for the Negative models (Mildly Negative and Strongly Negative) at 4.8 and 4.9 s, respectively. Once the

models are trained, however, the actual runtime of the model for testing sets is very short and is mostly proportional to the size of the test sets.

5.2. SVM sentiment analysis results

An SVM model was prepared using Thorsten Joachims' SVM Light application (Joachims, 1999). The SVM Light application ran in linear kernel mode with a 10 QP-sub problem setting. Similar to DAN2, a model was constructed for each individual sentiment class (using "one vs. all" approach). The 187 individual features that were selected for DAN2 were also used as the input to the SVM model. The format of the training data provided to SVM was ([positive example] [feature number]:[value] [feature number]:[value]...). An example of a training row associated with a mildly positive tweet is as follows:

11 : **146** : **153** : **1107** : **1166** : **1176** : **1180** : **1**

The SVM model was trained on 1500 records and tested against 1418 records. Similar to the DAN2 models, four sets of SVM models, one for each sentiment classes, were created. Tables 6 and 7 also present results of applying SVM to the same datasets. The recall values for SVM show only good performance for the "Strongly Positive" class (at 94.1%), the remaining three recall values for this model are relatively poor. Similarly to DAN2, the best accuracy values are for the two "Negative" classes (88.3% and 94.6%).

The SVM model appears to be the most susceptible to the similarities between the overlapping mildly positive and strongly positive features (exhibiting the least accuracy in selecting mildly positive records). The greater number of strongly positive records influenced the model to more frequently choose to place borderline records into the strongly positive category (hurting its accuracy in the mildly positive class).

5.3. Comparing DAN2 vs. SVM

Tables 6 and 7 present recall and accuracy values for both DAN2 and SVM using identical input values (Tweets and features). The SVM is one of the most widely used tools for sentiment analysis and we have selected it as the benchmark for this study. When comparing DAN2's recall and accuracy values with the corresponding values from SVM, DAN2 produces better results in recall across all sentiment classes and better accuracy in three of the four classes. When examining DAN2's improvement in recall on a class-by-class basis we note that the highest improvements are for the two "Mildly" categories. DAN2's performance improvement on the "Mildly Negative" category is more than a 100% improvement upon SVM's recall value. For the "Mildly Positive" category, SVM completely fails to assign any tweet to this class, so DAN2's performance improvement is even more pronounced. When comparing the two "Strong" categories, DAN2's performance improvements in recall for the "Strongly Negative" and "Strongly Positive" classes is 52.8% and 2.02%, respectively. To assess the statistical significance of these performance improvements provided by DAN2's over the benchmark SVM values, pair-wise t-tests were conducted ($n = 1418$; alpha = 0.05; two-tailed test). The results of these pair-wise *t*-tests are indicated in Tables 6 and 7 using daggers (†). In two of the four sentiment classes, the improvements in recall provided by DAN2 over SVM were statistically significant with $p < 0.10$. The improvements in recall provided by DAN2 were even more impressive in the "Mildly Positive" and "Mildly Negative" sentiment classes with significance at $p < 0.01$. The performances of DAN2 and SVM were closer in terms of accuracy, with three of the four DAN2 models outperforming the comparable SVM models. The SVM results show slightly better accuracy than DAN2 (2.29% improvement) for the "Strongly Positive" category, while DAN2 shows better accuracy for the other three categories. DAN2's best performance in accuracy compared to SVM is for the "Mildly Positive" category with 4.38% improvement. In the "Mildly Positive" and "Mildly Negative" sentiment classes the improvement in accuracy were statistically significant with $p < 0.01$ and $p < 0.10$, respectively. Overall, the results show DAN2 to be a very effective alternative for Twitter sentiment analysis as measured in this experiment.

Analysis of the results show that the SVM model struggles to identify the tweets associated with less frequently occurring categories (i.e. they could not find the needle in the haystack). The SVM models more often predicted that the tweet did not fit into the less frequently occurring category (i.e. they choose "haystack"). This approach, however, helped overall accuracy as more often than not the "haystack" was the correct choice. Favoring the statistical weight of the "haystack" allowed the SVM model to close the gap with the DAN2 model in terms of the accuracy metric, even though the accuracy for each sentiment class is less favorable. While DAN2 exhibited greater overall accuracy in three of the four sentiment classes it showed improvements in recall over the SVM approach for every class. DAN2 appears to be much better suited to the practical applications associated with sentiment analysis. By identifying strongly negative tweets with a recall rate that is significantly greater than that found with SVM (29.5% vs. 19.3%), a brand manager will have 53% more negative tweets to inform their marketing activities. While the strongly negative sentiment category is the least frequently occurring, it is also an important consideration for brand managers that are attempting to identify problems with brand image.

5.4. Twitter-specific lexicon effectiveness

The feature engineering approach introduced earlier resulted in a reduced Twitter-specific lexicon with only 187 features. Of the 187, six were found to be brand-specific (Bieber-specific) while the remaining 181 were generic to all Twitter messages. We also showed that one or more of the terms in the reduced lexicon set were still present in more than 97.3% of the entire Twitter corpus used in this research (10,345,184 Tweets). To show the effectiveness of the reduced Twitter-specific features for sentiment analysis, we developed SVM models using a traditional sentiment lexicon from the Opinion Finder system (Wilson et al., 2005) to define the input feature space. We selected Opinion Finder and SVM because these are widely used sentiment analysis tools applied by researchers and are readily available. The Opinion Finder lexicon is comprised of 7630 terms expressing positive or negative sentiments. Tables 6 and 7 present the results of running the SVM model with Opinion Finder's lexicon and the associated recall and accuracy values (columns SVM-OF Test). The results show that the reduced lexicon set prepared using our approach indeed offers more favorable values for recall and accuracy in three of the four sentiment classes. In particular the recall values for the Opinion Finder lexicon models are significantly lower than the corresponding values for the SVM models based upon the reduced lexicon set. We note that the SVM model based on the Opinion Finder's lexicon set was only effective in recall for the "Strongly Positive" category. Even for this category, the SVM model based on the Twitter-specific lexicon performance shows improvement of more than 33.8% in recall over its corresponding Opinion Finder value. Clearly, the ineffectiveness of the Opinion Finder's model for the two "Negative" categories is an additional evidence for the development of the Twitter-specific lexicon set. While the performances of the two lexicons are closer when comparing the results using the accuracy metric, our reduced lexicon produced improvements in three of the four sentiment classes, with the "Strongly Positive" category showing the most improvement (5.79%) over its corresponding

Opinion Finder's model. To assess the statistical significance of the performance improvements associated with application of the Twitter-specific lexicon to define the feature space for the SVM models in comparison with using the benchmark Opinion Finder lexicon, pair-wise t-tests were conducted ($n = 1418$; alpha = 0.05; two-tailed test). The results of these pair-wise t-tests are indicated in Tables 6 and 7 using asterisks ($*$). In three of the four sentiment classes, the improvements in recall provided by the devised lexicon were highly significant with $p < 0.01$. While the performances of the two lexicons were closer in terms of accuracy, in the "Strongly Positive" and "Strongly Negative" sentiment classes the improvement in accuracy were statistically significant with $p < 0.05$ and $p < 0.01$, respectively. Overall, the results show that the reduced Twitter-specific lexicon set is very effective for Twitter sentiment analysis as measured in this experiment. Finally, the choice of using widely available modeling tools (e.g. SVM and Opinion Finder) should enable other researchers to use the reduced feature set to further validate its effectiveness.

6. Conclusion

This research makes several contributions to Twitter sentiment analysis, demonstrated through application on a corpus of tweets related to the Justin Bieber brand. Earlier research on Twitter classification has classified factual sounding tweets as a neutral tweet (Go et al., 2009). Using this approach, they state that "more than 80%" of tweets contain no sentiment. Our approach to sentiment analysis has increased sensitivity, accounting for tweets with mild sentiment (positive and negative), resulting in a more accurate identification of the neutral category. Using our approach on the Justin Bieber Twitter corpus, we find the number of neutral cases to be closer to 10%. Additionally, where some previous investigations have limited or removed emoticons from consideration, this research finds emoticons to have high explanatory power. The effectiveness of the emoticons in our application may be in part due to the communication tendencies of users tweeting about Justin Bieber, however the length restriction imposed upon tweets encourages the shortening of such expressions making them an important and common practice in Twitter in general. Finally, we find the "retweet" class to be a distinct class, separate from direct sentiment and much more easily characterized in terms of sentiment.

Our process resulted in a more accurate estimation of sentiment in experimentation on the Justin Bieber Twitter corpus. To begin with, data gathering tools were developed that allowed for an extremely large number of tweets related to the brand, while scoring tools allowed for a large training and testing datasets. Next, our feature engineering approach resulted in a Twitter-specific lexicon with a reduced feature set with the ability to still characterize 97.3% of all messages in the 10,345,184 Justin Bieber Twitter corpus. The smaller, Twitter-specific lexicon reduces problem complexity (model size reduction), maintains a high degree of coverage over our Justin Bieber Twitter corpus, and yields improved sentiment classification accuracy in our experiments. The Twitter-specific lexicon developed in this study consisted of 181 terms of sentiment expression that are general to Twitter, and only 6 Justin Bieber brand-specific. Although our data and experimentation in this study has focused on a single brand, we expect future experiments using the Twitter-specific lexicon on other brand-related and general Twitter corpuses can result in similarly outstanding sentiment analysis performance. We acknowledge that research in Twitter sentiment analysis is still emerging and additional studies are required to be able to claim such definitive conclusions.

The experimentation on our Justin Bieber Twitter corpus also shows that the DAN2 model was much better at finding the messages of interest (messages that belonged in a specific sentiment class) at the cost of offsetting false positives. This favorable performance toward better recall is extremely desirable in a number of marketing and public relations use cases, including social media outreach, in which the most positive tweets need to be found for re-tweeting, and the worst tweets need to be uncovered for purposes of damage control. In both use cases the cost of reviewing a false positive far outweighs the cost of missing a brand-affecting tweet. DAN2's ability to greatly improve recall at the edges of sentiment classes (while at the same time nominally increasing overall accuracy) is extremely beneficial in the considered use cases. Our future research will examine the performance of DAN2 in sentiment analysis on other brand-related and more general Twitter corpuses, with similar results anticipated.

The use of a highly explanatory Twitter-specific lexicon in this research demonstrated that it offered capabilities for the Justin Bieber brand to recognize emerging issues with their brand identity. We expect future experimentation on additional brand-related and general Twitter corpuses to reveal similar findings. The approaches and tools developed in this research will allow brands to monitor key sentiment indicators on incoming tweets from users (i.e. follow request sentiment, re-tweet sentiment and message sentiment). These metrics, when used to influence brand decisions (brand offerings, frequency of brand messaging, timing of messaging, type of brand messaging, brand reactions to external factors) will allow brand managers to make better use of the Twitter service and best influence public perception.

References

Abbasi, A., Chen, H., & Salem, A. (2008). Sentiment analysis in multiple languages: Feature selection for opinion classification in web forums. *ACM Transactions on Information Systems, 26*(3), 1–34. 12.

Agarwal, A., Xie, B., Vovsha, I., Rambow, O., & Passonneau, R. (2011). Sentiment analysis of twitter data. In *Proceeding of ACL HLT conference* (pp. 30–38).

Aue, A., & Gamon, M. (2005). Customizing sentiment classifiers to new domains: a case study. In *Proceeding of the intl. conference on recent advances in natural language processing*. Borovets, BG.

Bakshy, E., Hofman, J., Mason, W., & Watts, D. (2011). Everyone's an influencer: Quantifying influence on Twitter. In *Proceeding of ACM WSDM conference Hong Kong*. China.

Barbosa, L., & Feng, J. (2010). Robust sentiment detection on twitter from biased and noisy data. In *Proceedings of the 23rd international conference on computational linguistics (COLING'10)* (pp. 36–44).

Benhardus, J., & Kalita, J. (2013). Streaming trend detection in Twitter. *International Journal on Web Based Communities, 9*(1), 122–139.

Bermingham, A., & Smeaton, A. (2010). Classifying sentiment in microblogs: Is brevity an advantage? In *Proceeding of ACM CIKM conference Toronto, Ontario, Canada* (pp. 1833–1836).

Bermingham, A., & Smeaton, A. (2011). On using twitter to monitor political sentiment and predict election results. In *Proceeding of IJCNLP conference, Chiang Mai, Thailand*.

Bifet, A., & Frank, E. (2010). Sentiment knowledge discovery in Twitter streaming data. In *Proceeding of 13th international conference on Discovery Science Conference* (pp. 1–15).

Blitzer, J., Dredze, M., & Pereira, F. (2007). Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceeding of the annual meetings of the association for computational linguistics* (pp. 440–447).

Bollen, J., Pepe, A., & Mao, H. (2011b). Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena. In *Proceedings of the fifth international aaai conference on weblogs and social media (ICWSM 2011)*, July, Barcelona, Spain, (pp. 1–10).

Bollen, J., Mao, H., & Zeng, X. (2011). Twitter mood predicts the stock market. *Journal of Computational Science, 2*(1), 1–8.

Cha, M., Haddadi, H., Benevenuto, F., & Gummadi, K. (2010). Measuring user influence in Twitter: The million follower fallacy. In *Proceeding of 4th AAAI conference on weblogs and social media*, Washington DC, (pp. 10–17).

Chung, J., Mustafaraj, E. (2011). Can collective sentiment expressed on twitter predict political elections? In *Proceedings of the twenty-fifth AAAI conference on artificial intelligence* (pp. 1770–1771).

Das, S., & Chen, M. (2007). Yahoo! for Amazon: Sentiment extraction from small talk on the web. *Management Science, 53*(9), 1375–1388.

Dave, K., Lawrence, S., & Pennock, D. (2003). Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *Proceeding of 12th intl. conference on the WWW*, (pp. 519–528).

Davidov, D., Tsur, O., & Rappoport, A. (2010). Enhanced sentiment learning using twitter hashtags and smileys. In *Proceeding of COLING conference'10, Beijing, China* (pp. 241–249).

Frakes, W. B., & Baeza-Yates, R. (1992). *Data structures and algorithms: Information retrieval*. Prentice Hall.

Gamon, M. (2004). Sentiment classification on customer feedback data: Noisy data, large feature vectors, and the role of linguistic analysis. In *Proceeding of the 20th intl. conference on computational linguistics* (p. 84).

Ghiassi, M., & Burnley, C. (2010). Measuring effectiveness of a dynamic artificial neural network algorithm for classification problems. *Expert Systems with Applications, 37*(4), 3118–3128.

Ghiassi, M., Olschimke, M., Moon, B., & Arnaudo, P. (2012). Automated text classification using a dynamic artificial neural network model. *Expert Systems with Applications, 39*(12), 10967–10976.

Ghiassi, M., & Saidane, H. (2005). A dynamic architecture for artificial neural network. *Neurocomputing, 63*, 397–413.

Ghiassi, M., Saidane, H., & Zimbra, D. K. (2005). A dynamic artificial neural network model for forecasting time series events. *International Journal of Forecasting, 21*, 241–362.

Gleason, B. (2013). #Occupy wall street: Exploring informal learning about a social movement on Twitter. *American Behavioral Scientist*.

Go, A., Bhayani, R., & Huang, L. (2009). Twitter sentiment classification using distant supervision. *Technical report, Stanford Digital Library Technologies Project*. (pp. 1–6).

Hsieh, C., Moghbel, C., Fang, J., & Cho, J. (2013). Experts vs. the crowd: examining popular news prediction performance on Twitter. In *Proceedings of ACM KDD conference, Chicago, USA*.

Hu (2013). Real-time Twitter sentiment toward thanksgiving and christmas holidays. *Social Networking, 2*, 77–86.

Huang, S., Peng, W., Li, J., & Lee, D. (2013). Sentiment and topic analysis on social media: A multi-task multi-label classification approach. In *Proceedings of ACM webSci conference, Paris, France*.

Jansen, B., Zhang, M., Sobel, K., & Chowdury, A. (2009). Twitter power: Tweets as electronic word of mouth. *Journal of the American Society for Information Science and Technology, 60*(11), 2169–2188.

Jiang, L., Yu, M., Zhou, M., Liu, X., & Zhao, T. (2011). Target-dependent Twitter sentiment classification. In *Proceeding of ACL conference* (pp. 151–160).

Joachims, T. (1999). Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, & A. Smola (Eds.), *Advances in Kernel methods-support vector learning* (pp. 1–56). MIT-Press.

Kajanan, S., Shafeeq Bin Mohd Shariff, A., Datta, A., Dutta, K., & Paul, D. (2011). Twitter post filter for mobile applications. In *Proceedings of the 21st workshop on information technology and systems* (pp. 1–6).

Kim, S., & Hovy, E. (2004). Determining the sentiment of opinions. In *Proceeding of the Intl. Conference on Computational Linguistics, Geneva, Switzerland* (pp. 1–8).

Kouloumpis, E., Wilson, T., and Moore, J. (2011). Twitter sentiment analysis: The good the bad and the OMG! In *Proceeding of AAAI conference on weblogs and social media*, (pp. 538–541).

Lerman, K., & Ghosh, R. (2010). Information contagion: An empirical study of the spread of news on digg and twitter social networks. In *Proceeding of 4th international AAAI conference on weblogs and social media, Washington DC* (pp. 90–97).

Liu, K., Li, W., & Guo, M. (2012). Emoticon Smoothed language models for Twitter sentiment analysis. In *Proceedings of AAAI conference*.

Loughran, T., & McDonald, B. (2011). When is a liability not a liability? textual analysis, dictionaries, and 10-Ks. *The Journal of Finance, 66*(1), 35–65.

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. Cambridge University Press.

Mathioudakis, M., & Koudas, N. (2010). TwitterMonitor: Trend detection over the twitter stream. In *Proceeding of ACM SIGMOD conference* (pp. 1155–1158).

Mejova, Y., Srinivasan, P., & Boynton, B. (2013). GOP primary season on Twitter: "Popular" political sentiment in social media. In *Proceedings of ACM WSDM conference, Rome, Italy*.

Mitchell, T. (2005). Reducing data dimension. machine learning 10-701, Carnegie Mellon University, http://www.cs.cmu.edu/~guestrin/Class/10701-S05/slides/dimensionality.pdf,1-40.

Mittal, A. & Goel, A. (2012). Stock prediction using twitter sentiment analysis. *Stanford University Working Paper*.

Naveed, N., Gottron, T., Kunegis, J., & Alhadi, A. (2011). Bad news travel fast: A content-based analysis of interestingness on Twitter. In *Proceeding of 3rd ACM WebSci conference, Koblenz, Germany*.

O'Connor, B., Balasubramanyan, R., Routledge, B., & Smith, N. (2010). From tweets to polls: Linking text sentiment to public opinion time series. In *Proceeding of 4th AAAI Conference on Weblogs and Social Media, Washington DC* (pp. 122–129).

Pak, A., & Paroubek, P. (2010). Twitter as a corpus for sentiment analysis and opinion mining. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)* (pp. 1320–1326).

Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the conference on empirical methods in natural language processing* (pp. 79–86).

Petrovic, S., Osborne, M., & Lavrenko, V. (2010). Streaming first story detection with application to Twitter. In *Proceeding of NAACL conference* (pp. 181–189).

Phelan, O., McCarthy, K., & Smyth, B. (2009). Using Twitter to recommend real-time topical news. In *Proceeding of ACM RecSys conference* (pp. 385–388).

Ringsquandl, M. & Petkovic, D. (2013). Analyzing political sentiment on Twitter. In *Proceedings of aaai conference*.

Romero, D., Meeder, B., & Kleinberg, J. (2011). Differences in the Mechanics of information diffusion across topics: Idioms, political hashtags, and complex contagion on Twitter. In *Proceeding of the 20th ACM International World Wide Web Conference on, Hyderabad, India*.

Rui, H., Liu, Y., & Whinston, A. B. (2011). Whose and what chatter matters? The Impact of Tweets on Movie Sales. NET Institute Working Paper No. 11-27, October, (pp. 1–30).

Saif, H., He, Y., & Alani, H. (2011). Semantic smoothing for twitter sentiment analysis. In *Proceeding of International Semantic Web Conference (ISWC)*.

Saif, H., He, Y., & Alani, H. (2012). Alleviating data sparsity for twitter sentiment analysis. In *Proceeding of the 21st ACM International World Wide Web Conference, Lyon, France* (pp. 2–9).

Tan, S., Wu, G., Tang, H., & Cheng, X. (2007). A novel scheme for domain-transfer problem in the context of sentiment analysis. In *Proceeding of the 16th ACM conference on information and knowledge management (CIKM)* (pp. 979–982).

Tetlock, P. (2007). Giving content to investor sentiment: The role of the media in the stock market. *Journal of Finance, 62*(3), 1139–1168.

Thelwall, M., Buckley, K., & Paltoglou, G. (2011). Sentiment in Twitter events. *Journal of the American Society for Information Science and Technology, 62*(2), 406–418.

Tumasjan, A., Sprenger, T., Sandner, P., & Welpe, I. (2010). Predicting elections with Twitter: What 140 characters reveal about political sentiment. In *Proceedings of the fourth international AAAI conference on Weblogs and social mediaa* (pp. 178–185).

Turney, P. (2002). Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics* (pp. 417–424).

Wang, H., Can, D., Kazemzadeh, A., Bar, F., & Narayanan, S. (2012). A system for real-time twitter sentiment analysis of 2012 U.S. presidential election cycle. In *Proceedings of ACL conference, Jeju, Republic of Korea*.

Wilson, T., Hoffman, P., Somasundaran, S., Kessler, J., Wiebe, J., Choi, Y., Cardie, C., Riloff, E., and Patwardhan, S. 2005. OpinionFinder: A system for subjectivity analysis. In *Proceedings of conference on human language technology and empirical methods in natural language processing* (pp. 34–35).

Wong, F., Sen, S., & Chiang, M. (2012). Why watching movie tweets won't tell the whole story? In *Proceedings of ACM workshop on online social networks* (pp. 61–66).

Yang, Y., & Pedersen, J. O. (1997). A comparative study on feature selection in text categorization. In *Proceedings of ICML-97, 14th international conference on machine learning* (pp. 412–420).

Zhang, L., Ghosh, R., Dekhil, M., Hsu, M. & Liu, B. (2011). Combining Lexicon-based and Learning-based Methods for Twitter Sentiment Analysis. *Hewlett-Packard Labs Technical Report HPL-2011-89*.