2015 Senior Thesis Project Reports

Iliano Cervesato* **Kemal Oflazer***
Mark Stehlik* **Khaled Harras***
Houda Bouamor* **Alexander W. Cheek***

May 2015
CMU-CS-QTR-127

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

*Qatar campus.

The editors of this report include the members of the Senior Thesis Committee on the Qatar campus and the students' advisors.

Abstract

This technical report collects the final reports of the undergraduate Computer Science majors from the Qatar Campus of Carnegie Mellon University who elected to complete a senior research thesis in the academic year 2014–15 as part of their degree. These projects have spanned the students' entire senior year, during which they have worked closely with their faculty advisors to plan and carry out their projects. This work counts as 18 units of academic credit each semester. In addition to doing the research, the students presented a brief midterm progress report each semester, presented a public poster session in December, presented an oral summary in the year-end campus-wide Meeting of the Minds and submitted a written thesis in May.

Keywords: Sentiment Analysis, Twitter, Machine Learning, Supervised Text Classification, Data Visualization.

Contents

State-of-the-art, Real-time, Interactive

Twitter Sentiment Analysis

using Machine Learning

AUTHOR

Sabih Bin Wasi

ADVISORS

Kemal Oflazer, Alexander W. Cheek

Acknowledgements

I wish to offer my gratitude to the advisors of this work, Prof. Kemal Oflazer and Prof. Alexander Cheek for letting me use their experience in building this system. It is to their credit that this undergraduate research turned out to be *state-of-the-art*. I would also like to give due credit to Prof. Behrang Mohit who initiated this research as part of the *Text Processing class* and encouraged to pursue this project beyond a course.

I would like to thank Dr. Houda Bouamor, Dr. Francisco Guzman and Prof. Thierry Sans who guided me selflessly throughout this research. My weekly meetings with Dr. Bouamor ensured steady progress of this work and helped me achieve the high system performance on time. Also, I am grateful to my friends Rukhsar Neyaz, Afroz Aziz and Abdullah Zafar for motivating me throughout.

This work wouldn't have been possible without the support of my parents who allowed me to fly away from home and always encouraged me to participate in research activities.

Abstract

With more than 500M tweets sent per day containing opinions and messages, Twitter has become a gold-mine for organizations to monitor their brand reputations and analyze their performance and public sentiments about their products. In this work, we present our efforts in building a machine learning based system for sentiment analysis of Twitter messages. Our system takes as input an arbitrary tweet and assigns it to one of the following classes that best reflects its sentiment: positive, negative or neutral.
We adopt a supervised text classification approach and use a combination of published-proven features and novel features. Our system outperforms the state-of-the-art systems, especially those tested on the SemEval 2014 testset. Using our prediction model, we also build an interactive visualization tool that uses standard design principles to provide interpretation and trust to the big-data fed into the system. Our system can handle Twitter sentiment analysis of millions of Tweets in pseudo real-time. Furthermore we give an extra emphasis to the adaptive nature of the tool in order to enable brand managers dig deep into patterns found in visualizations -- resulting in effective public sentiment monitoring on social media.

Table of Contents

1 Introduction

With an exponential rise in social media usage to share emotions, thoughts and opinion, Twitter has become the gold-mine to analyze brand performance. Opinions found on Twitter are casual, honest and informative than what can be picked from formal surveys etc. Millions of users express their sentiment about brands they interact with. These sentiments, if identified, can be useful for companies to not only monitor their brand's performance but also locate aspects and time periods that receive polar sentiments. These brands can be products, celebrities, events or political parties. Therefore, teams were dedicated to analyze candidate's performance during national polls[1], or to analyze public response during the launch of a gadget or a movie.

However, with more than 500M tweets sent per day[2], this data has already become huge enough to be analyzed by any team manually. Likewise, the diversity of tweets presumably cannot be captured by fixed set of rules designed by hand. It is worth noting that the task of understanding the sentiment in a tweet is more complex that of any well-formatted document. Tweets do not follow any formal language structure, nor they contain words from formal language (i.e. out-of-vocabulary words). Often, punctuations and symbols are used to express emotions (smileys, emoticons etc).

In this work, we use machine learning approach and natural language processing techniques to understand the patterns and characteristics of tweets and predict the sentiment (if any) they carry. Specifically, we build a computational model that can classify a given tweet as either positive, negative or neutral based on the sentiments it reflects. A positive and negative class would contain polar tweets expressing a sentiment. However, a neutral class may contain an objective or subjective tweet - either a user reflect neutrality in an opinion or contain no opinion at all. Examples of each class can be found in Table 1. The decision to use three classes is made to accommodate the complexity of the problem and is consistent with ongoing research in the field. The experiments conducted on our sentiment predictor informally shows that our system is amongst the best performing systems for this task.
Using our sentiment predictor, we also build an interactive-visualization tool to help businesses interpret and visualize public sentiments for their product and brands. This tool enables the user to not only visualize plain sentiment distribution over the entire dataset, but also equips user to conduct sentiment analysis over the dimension of time, location, and influencing power[3] of a user.

Class	**Tweet**
positive	@hon1paris: I <3 1D too! #muchlove
negative	The new Transformers suck!! Wasted my time and money!!!
neutral	Well, I guess the govt did what it could. More needed though! I plan to wake up early in the morning #early2bad

Table 1: Sample tweets for each sentiment class

This work is structured as follows: we first discuss our procedure for building machine learning model and explain the basics of machine learning technique used for text classification problems (Section 2). Then, we

[1] http://mprcenter.org/blog/2013/01/how-obama-won-the-social-media-battle-in-the-2012-presidential-campaign/
[2] https://about.twitter.com/company
[3] We deduce influencing power of a twitter user by their number of followers.

summarize the findings from the literature review conducted to understand the research field and to identify gaps in knowledge. Once these gaps are identified, we lay-out the details of our system and our motivation in choosing the features for this system. We then run experiments on this system to compare the performance of our system with other published work. Also, the effectiveness of each feature added to the system is highlighted. Lastly, our research contributions are summarized with future steps needed to be taken (Section 3).

The second section discusses our procedure for building interactive visualization tool that uses sentiment prediction model. We summarize the findings from the literature review conducted to understand what metrics and features have worked well and how can they be adopted to sentiment analysis visualization. We also describe other related public tools that perform the similar task. After highlighting past work, we describe our approach and system architecture for developing the visualization tool. Next, we present the screenshots of the application describing the motivation for each of user view and element. Lastly, we summarize our contributions and future steps needed.

2 Machine Learning Background

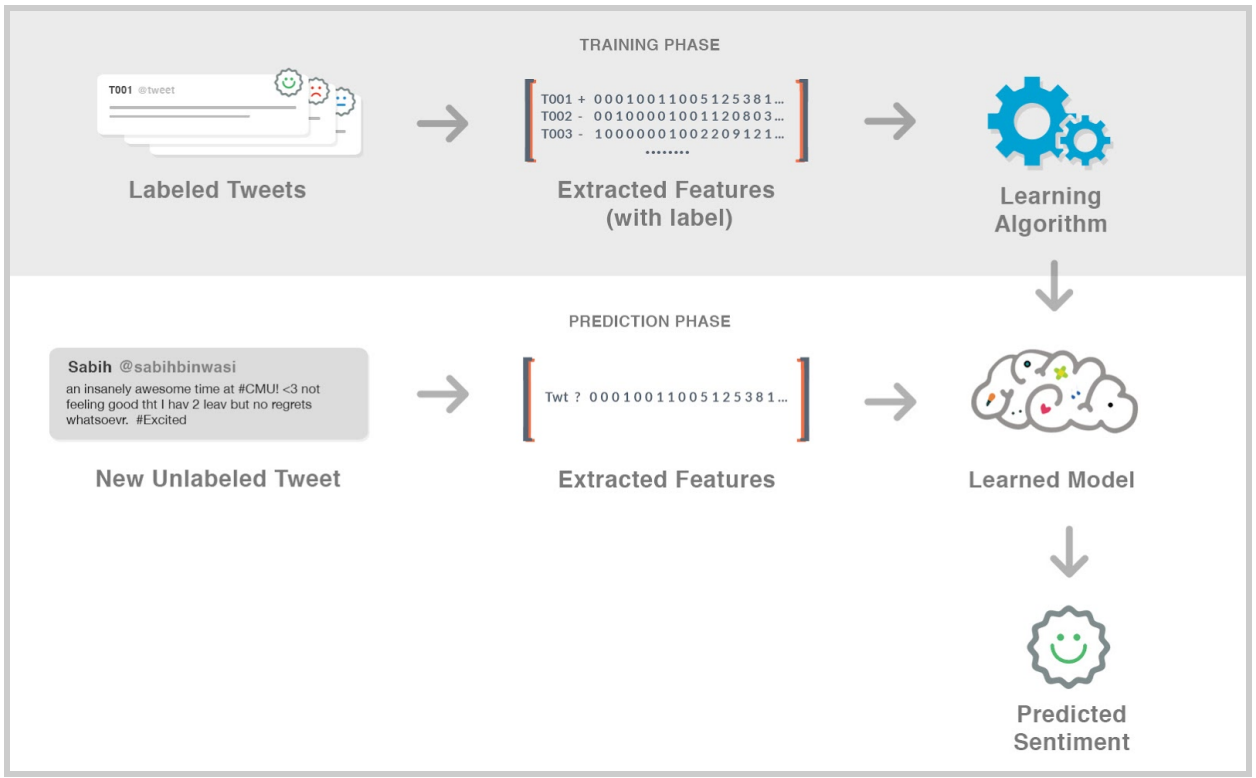


Figure 1: Overview of Supervised Sentiment Classification of Tweets

Before understanding research conducted for Twitter sentiment analysis, we need to describe standard procedure to tackle this problem. Supervised Text Classification, a machine-learning approach where a class-predictor is inferred from labeled training data, is a standard approach for sentiment classification of tweets. The overview of this approach, adapted to sentiment classification of a tweet, is illustrated in figure 1. First, a dataset of labeled tweets are compiled. Each tweet in this set is labeled positive, negative or neutral by a human annotation based on the sentiment the annotator(s) concluded after analyzing the tweet.

Then, a feature extractor generates a feature vector for each labeled tweet where the values of this vector should characterize the sentiment. Once feature vectors are extracted for each tweet in the labeled dataset, they are fed to classification algorithm that attempts to find relations between each value (called feature) in the vector and the labeled sentiment. Popular classification algorithms used for this task are SVM (support vector machines), Naive Bayes and Maximum Entropy method. Studies have compared several classification algorithms and highlighted the aforementioned algorithms as most-effective (Pang et al., 2002). The feature-sentiment relationship is captured by these algorithms and stored in a learned model. When new instance is provided to the model, it uses already-learned relationships to predict the sentiment.

3 Literature Review

One of the first popular work in the field of Twitter sentiment analysis was done by Go et al. (2009). The work was primarily conducted as a project work that turned into research publication. The research uses distant supervision technique to overcome the problem of manual annotation of large set of tweets. Tweets with ":)" emoticon were considered positive while tweets with ":(" in the message were considered negative. This resulted in two defects. First, emoticons, which were considered important by other works in the area (see below) couldn't be used to learn sentiments. Secondly, the authors are unsure if all tweets with ":)" are truly positive or can contain negative or sarcastic sentiments too. Therefore, the dataset used in the experiment was labelled noisy. The task of classification was limited to positive and negative though the need for neutral class was highlighted in the end. Go et al. (2009) used SVM, MaxEnt and Naive Bayes and reported that SVM and Naive Bayes were equally good and beat MaxEnt. The research also found POS (parts of speech) tags were not helpful for their purpose.[4] However, they only tried general purpose POS tagger which is shown to have inaccuracies when ran on microblogging text.

Another popular work in this topic was conducted by Pak and Paroubek (2010). The same distant supervision procedure of tagging tweets positive or negative based on the emoticons it mentions was used. However, highlighting the significance of neutral class, the team also collected neutral tweets. These tweets were strictly objective and were collected from newspapers and magazines. Unlike Go et al. (2009), POS features were deemed useful. However, TreeTagger (Schmid, 1995) was used which wasn't designed to work with microblogging text like tweets.[5] The research didn't use auto annotated tweets in the test set. Instead, a small size test set was hand annotated (216 tweets). Like Go et al. (2009), Pak and Paroubek (2010) also used linear kernel SVM to run the experiment. The results were not reported as accuracy metric reported by Go was different and hence the results were not comparable to previous research work.

Major boost in the discussed field came as a result of annual SemEval workshop (Nakov et al., 2013).[6] The workshop had more than 30 entries in 2013 when the organizers first introduced twitter sentiment analysis as one of their exercises. Not only this provided opportunity for more frequent research in the area, it provided large dataset of hand annotated tweets with all three sentiment classes - positive, negative and neutral (see Section 4.1). The consistent scoring metric was also available for the community to gauge their research with other teams in the field.

[4] Parts of speech tags are labels assigned to a word in a text based on its grammatical context. For example, in a text "Apples are red", the respective POS tags for words would be noun, verb and adjective.
[5] TreeTagger is a tool for annotating text with part-of-speech and lemma information.
[6] https://www.cs.york.ac.uk/semeval-2013/task2/

The best performing system in SemEval 2013 workshop was by Mohammad et al. (2013). The work utilized the hand annotated datasets by workshop organizers and build a classifier that does the same job as the tool built in this project. Unlike previous work, this system made use of POS tagger and tokenizer designed specifically for Twitter (Gimpel et al., 2011). The feature set included a large number of features with some focusing on the specific nature of tweet like the use of elongated words, emoticons, all-caps words, url link etc. They also relied heavily on lexicons that assigned a sentiment to a token as a positive real number (if a positive word) or negative. These features are called lexical features. It was reported that after bag of words (that almost all research works have used), lexical features were most useful. Like the aforementioned research work, a linear kernel SVM was used. The results are published using the metric provided by workshop organizers: the average F1 measure of positive and negative class (see Section 5.1). With all features included, the system received the score of 69.02 on Mohammad et al. (2013) provided testset.

The best performing system in the second run of SemEval workshop (Rosenthal et al., 2014) was by TeamX (Yasuhide, 2014). Building on the system reported by Mohammad et al. (2013), this system also relied on lexical features. The system introduced the concept of cost-sensitive classification in the task of Twitter sentiment classification. They found the training data to be unbalanced, biasing the learner towards neutral class. To account for this bias, the paper proposed to assign higher penalty for misclassification of polar classes. This was reported to nullify the bias and increase overall accuracy.

In this second run of SemEval workshop, we also submitted our system (Bin Wasi et al. ,2014) that we built as part of a mini course project (15-383: Introduction to Text Processing). Our approach was similar to Mohammad et al. (2013) but differs in the combination of features used to build the feature vector. The reasoning to choose a smaller feature vector was to only add features that showed significant improvement to the baseline. Our system achieved a score of 65.11 on the test set provided in the workshop.

Looking at the broader field of sentiment analysis of any text, results reported by Pang et al. (2002) were insightful for our research. First, they compared different machine learning algorithms to suggest which classification algorithms work better for this type of text categorization and reasoning for their better performance. Also, the challenges posed by the sentiment classification were highlighted which helped us understand this task better and then building the right feature vector for our system.

In summary, we reviewed numerous papers that highlighted the positive impact of lexical features. Each system used their own methodology to represent these lexical features in the feature vector. We noted that most papers combined these lexical features into one score that was added as a value in the feature vector. There wasn't enough work that explored the idea of converting this lexical analysis into numerous features that captures the independent nature of lexical features and let the classification algorithm deduce the relationship between these features using the training data. We explored this promising area of lexical features in-depth through this research. Also, unlike research done earlier, this work uses a very conservative approach in adding features to the system. We also build on Yasuhide (2014) ideas about cost-sensitive classification and identify areas of improvement. Most of the remaining aspects of our system were inspired by earlier reported works that are cited in this section.

4 Methodology

We developed sentiment analysis system using the standard machine learning approach as explained in the background section.

4.1 Labeled Data

We use SemEval 2014 provided data for our research (Rosenthal et al., 2014). The datasets were carefully created by SemEval 2014 organizers over a period of one year. Diverse categories of tweets like politics, chat tweets and celebrity fan interactions were included. They report that the tweets represent large portion of twitter specific vocabulary like popular hashtags and mentions. The collected dataset was then annotated by hand on Mechanical Turk. Annotators were given guidelines from organizers as to what should be considered positive, negative or neutral to minimize the subjective behavior at sentiment annotation. The publishers highlighted that the dataset was cleansed from outliers. The publishers further stated that class distribution was reflective of real world tweets. Though intuitively the argument makes sense (neutral being the majority), we didn't spend effort to scientifically validate the claim. The data containing 9685 sentiment-labeled tweets was published on the workshop website. Table 2 shows the class distribution of this dataset.

Labeled Sentiment	Share
Positive	38%
Neutral	47%
Negative	15%

Table 2: Class distribution in training dataset

SemEval 2014 also released a labeled development dataset which is used for in-depth analysis and internal experiments. This dataset contained 1,655 tweets with class distribution similar to that of the training data. Using development data to identify useful features and better-performing algorithm is standard practice in machine learning field. Since this data is looked at often by researchers and systems are adjusted to perform well on this set, it is essential that development set does not overlap with the training dataset. This was ensured by Rosenthal et al. (2014).

We conducted a qualitative analysis to explore our development set. We found some instances that were annotated arguably incorrect. For example, the following tweet was marked neutral whereas the text conveys positive sentiment:

> *@_Nenaah oh cause my friend got something from china and they said it will take at least 6 to 8 weeks and it came in the 2nd week :P*

We consider this instance (marked negative) as noise in the dataset. Nevertheless, the dataset is significantly better than other datasets which were automatically annotated using distance supervision (see Literature Review section).

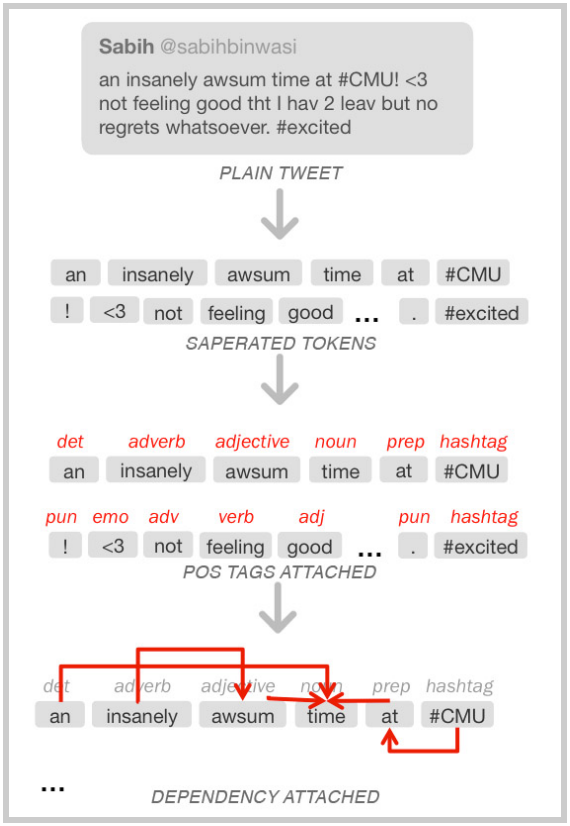
4.2 Preprocessing

Before the feature extractor can use the tweet to build feature vector, the tweet text goes through preprocessing step where the following steps are taken. These steps convert plain text of the tweet into processable elements with more information added that can be utilized by feature extractor. For all these steps, third-party tools were used that were specialized to handle unique nature of tweet text. These tools were developed by a research group at Carnegie Mellon University (Gimpel et al., 2011; Kong et al., 2014). All three preprocessing are illustrated in Figure 2.

Step 1: Tokenization

Tokenization is the process of converting text as a string into processable elements called tokens. In the context of a tweet, these elements can be words, emoticons, url links, hashtags or punctuations. As seen in Figure 2, “an insanely awsum….” text was broken into “an”, “insanely”, “awsum”...

These elements are often separated by spaces. However, punctuation ending the sentence like exclamation marks or full-stop are often not separated by a space. On the other hand, hashtags with “#” preceding the tag needs to be retained since a word as a hashtag may have different sentiment value than a word used regularly in the text. Therefore, Twitter-specific Tokenizer [8] is used to extract tokens.



Step 2: Parts of Speech Tags

Parts of Speech (POS) tags are characteristics of a word in a sentence based on grammatical categories of words of a language. This information is essential for sentiment analysis as words may have different sentiment value depending on their POS tag. For example, word like “good” as a noun contains no sentiment whereas “good” as an adjective reflect positive sentiment.

As seen in Figure 2, each token extracted in the last step is assigned a POS tag. Like in Step 1, we use Twitter-specific POS tagger (Gimpel et al., 2011) that can handle out-of-vocabulary words (OOV) and Twitter related tags like hashtags and emoticons. The accuracy of this POS tagger is 93%.

Step 3: Dependency Parsing

For our purposes, dependency parsing is extracting the relationship between words in a sentence. This can be useful in identifying relationship between “not” and “good” in phrases like “not really good” where the relationship is not always with the adjacent word. The dependency parsing tool we use (Kong et al., 2014) gave parent-child relationship between tokens in a tweet as seen in Figure 2. The accuracy of the dependency parser is 80% on Twitter data.

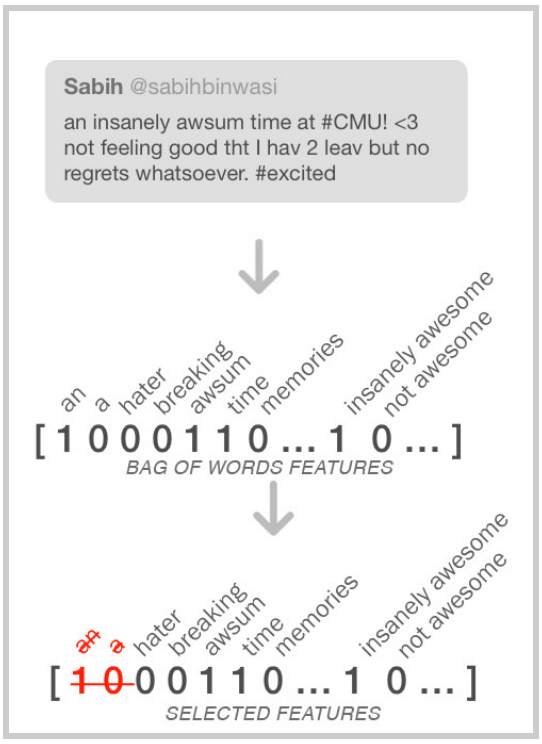
4.3 Feature Extraction

Feature extraction is the process of building a feature vector from a given tweet. Each entry in a feature vector is an integer that has a contribution on attributing a sentiment class to a tweet. This contribution can vary from strong, where the value of a feature entry heavily influence the true sentiment class; to negligible, where there is no relationship between feature value and sentiment class. It is often the job of classification algorithm to identify the dependency strength between features and classes, making use of strong correlated features and avoiding the use of 'noisy features'.

During this work, we followed incremental feature selection approach. After adding each set of features, we conducted experiments to compare the performance of the system before and after the feature addition. If the feature set significantly improved the performance of the system, it remained in the feature vector; otherwise, it was removed.
Performance was evaluated using a standard cross-validation method. The training dataset was pseudo-randomly divided into ten non-overlapping subsets preserving the class distribution. The accuracy of the system was evaluated using nine folds as the training set and one fold as the evaluation set. This process occurs ten times, each time selecting a different fold as evaluation set. Once all ten accuracy scores are collected, an average is calculated that we use as a performance metric.

4.3.1 Bag of Words Feature Set

Bag of Words (unigrams) is a set of features where the frequency of tokens (or in our case, presence of a token) is indicated in a feature vector. From our study of past work, this feature set was unanimously chosen by researchers to be included in the feature vector. An entry in the feature vector is assigned to each unique token found in the labeled training set. If the respective token occurs in a tweet, it is assigned a binary value of 1 otherwise it is 0. Note that the grammar structure or ordering of token sequence is not preserved. Instead, only the independent presence of a token preserved. As seen in Figure 3, token "awsum" occurs in a given tweet and hence that column gets a value of 1. On the other hand, the word "hater" does not occur in a tweet and therefore has a value of 0. Note that, "hater" would have occurred in some tweet instance in the training dataset since we have a column for it in a feature vector.



The effectiveness of using bag of words for sentiment analysis have been reported by various publications (Pang et al., 2002; Mohammad et al., 2013; Bin Wasi et al., 2014). It was also found that indicating only presence of a word yields higher performance than indicating the frequency of a word. The underlying cause for this behavior may be that the sentiment class is not usually varied if certain words occur more than once in the text.

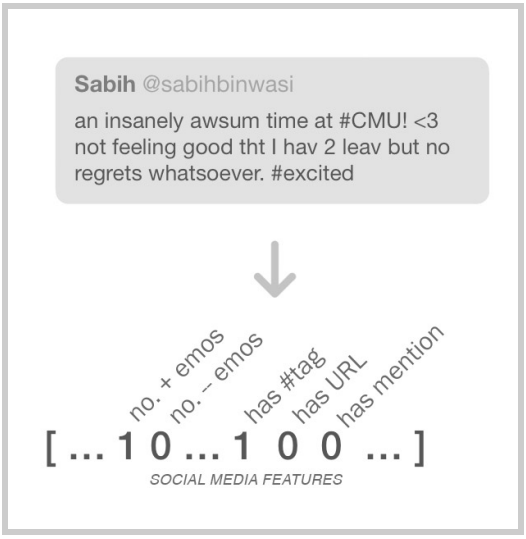
Apart from assigning an entry for each unique token in training dataset, we also assign an entry for each unique ordered pair of tokens (bi-grams) found in the dataset. As seen in Figure 3, “insanely awsum” is assigned an entry where the value is 1 if that pair occurs in a tweet 0 otherwise. This equips the system to not only indicate the presence of a token but also its context. For example, having an entry “not awsum” in a feature vector would help the classification algorithm to trust that entry value more than the value of “awsum” if the pairs occurs in the text. We extended our feature vector to also include tri-grams (ordered sequence of three tokens) to capture further context.

4.3.2 Feature Selection

As one can expect, adding unigrams, bigrams and trigrams adds large number of entries to a feature vector. In our case where we had 9685 tweets, this feature set resulted in more than 300,000 feature entries. This made the vector space high dimensional making the task of identifying relationship between each dimension (i.e. feature) much more complex and slow. This problem is common in the task of text classification and is referred to as Curse of Dimensionality. We noticed that most features in this set were not relevant for the task of sentiment analysis and hence we decided to remove some of these features. Past work have identified features that occur less than a specific value selected and optimized carefully in the training set and removed them. The idea is that tokens that do not occur frequently in the training set may be rare and hence have less importance for the task.

It is worth noting that words that may occur infrequently can still heavily influence the sentiment of a tweet. For example, we noticed that “#7FactsAboutMyBestFriend” occurs only once but can still single-handedly influence the sentiment of a tweet. Therefore, we conducted a feature attribution evaluation to identify features that have an impact. A popular attribute evaluation method Chi-Squared Feature Selection was used. This method runs a classification algorithm and evaluates the dependency between a feature value and each class. If the feature is found to have a higher dependence (correlation), it is assigned a higher rank. Kavitha et al. (2012), evaluating alternative evaluation methods, showed that for text classification, Chi-Squared performed the best.

Once all the features are assigned a rank, we picked the top 1800 features and removed all other features from the feature vector. The parameter value of 1800 was picked using standard one-fold-out cross-validation parameter selection method where performance on different dimension values (we used 500 to 2500, equispaced by 100) were evaluated and the best performing parameter value was chosen. This method attempts to avoid over-fitting the parameter value to one particular dataset and maximizes the chances of performing well on an unseen dataset. We also use this method for choosing other parameter values (see Section 4.5).



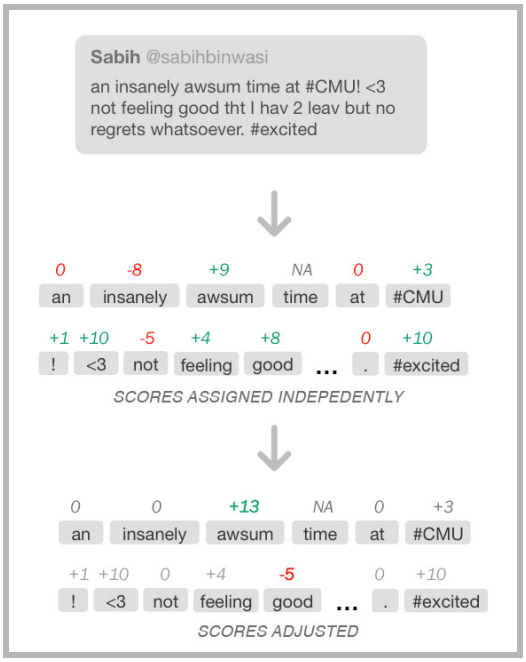
4.3.3 Social Media Feature Set

Emoticons are symbols that acts like an expression of emotion using punctuation and language characters. For example, in Figure 4, “<3” is an emoticon that depicts heart which usually represent *love*. These emoticons are often strongly correlated to either positive or negative sentiment unless used in the context of sarcasm. Therefore, some form of indication for the presence of emoticon is indicated in the feature vector of various published work (Mohammad et al., 2013, Bin Wasi et al., 2014; Yasuhide, 2014). Building on these reported works, we added the number of positive and negative emoticon in a tweet as two additional features in the vector. This value was generated using a dictionary of positive and negative emoticons we built using Wikipedia listing of emoticons with their representative emotions (see Appendix A).[7] For the sample tweet illustrated in Figure 4, number of positive emoticons found as per our dictionary is 1 while no negative emoticons are found.

Additionally, the presence of hashtag, URL and mention (“@username”) is also indicated in the feature vector. Each of these features have been used in past work showing some impact in differentiating between polar (positive and negative) and non-polar (neutral) class. This may be because these indicators increase the chances of identifying relationship between them and a sentiment class. For example, it may be the case that tweets containing a mention are subject to polar classes. However, since a tweet with “@username1” is a rare token in the training dataset, the classification algorithm would fail to identify this relationship. With only “@” as a binary feature, the chances for that relationship discovery increases.

4.3.4 Lexical Feature Set

Lexical feature sets refer to features that are driven by the use of lexicons. In our task, sentiment lexicons are popular that map tokens or n-grams to a polarity score. Their usage in published works (Mohammad et al., 2013, Bin Wasi et al., 2014; Yasuhide, 2014) have been reported successful due to their ability to localize the problem of sentiment classification. After bag-of-words features, Mohammad et al., 2013 report this feature set to be most successful. To solve the problem of identifying the sentiment of full text, sentiment of a token is identified using lexicons which is a much easier task.



Our system primarily uses AFINN lexicon (Nielsen, 2011). This lexicon was manually constructed to map frequently used polar words on Twitter to the polarity score in range [-5 and +5] where -5 is assigned to a very negative token and +5 represent very positive token. Tokens containing no sentiment by themselves are assigned the score of 0. Each token in a tweet is assigned in a polarity score independently. It is worth noting that the AFINN lexicon contains 2477 tokens - resulting in some tokens to not get any score assigned. For example in Figure 5, “time” is not assigned any score.

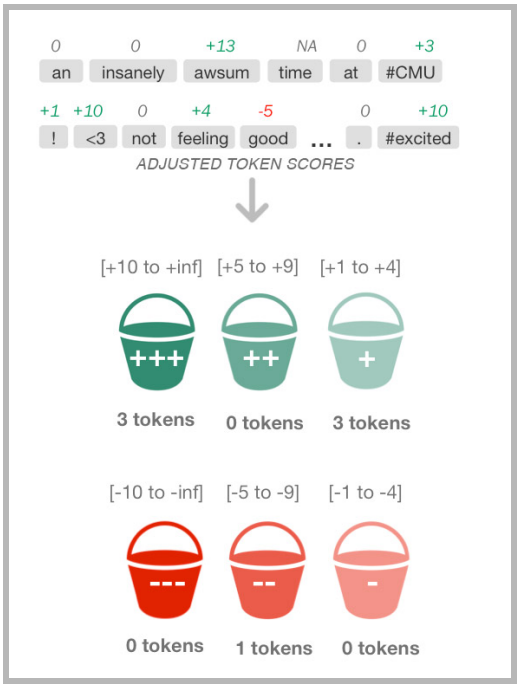
[7] http://en.wikipedia.org/wiki/List_of_emoticons

4.3.4.1 Handling Negations and Intensifiers

The problem highlighted in the past regarding the usage of sentiment lexicon is that these lexicons fail to capture the context of a word (Zhu et al., 2014). As seen in Figure 5, “good” is assigned a positive score of +8. However, in the context of “not feeling good”, good should be assigned a negative score. This is referred to as negation context problem. Additionally, “insanely” and “awsum” are assigned a negative and positive score respectively. In the context of “insanely awsum”, “insanely” is not used independently but used to intensify the sentiment of “awsum”. This is referred to as intensifier context problem.

To deal with negation problem, we built on the work of Councill et al. (2010) in which the authors identify words that create negation context. We use that list, together with dependency tree parsing (see Section 4.2), to readjust the scores. Every time a word that can create negation (not, neither, never etc.) is found, the system track its head word token. This negated token was mostly found a level above the negation word. The score of the negated word was then readjusted. Unlike past work where the scores are flipped if negation context was detected, we also reduced the weight of a token. While studying the adjusted scores, we found that negated tokens were assigned absolute higher scores compared to other tokens with similar sentiment. Specifically, if “bad” and “good” are assigned same absolute score, “not good” would have the same polarity score post-negation adjustment as “bad”. However, “not good” didn’t reflect as strong of a negative sentiment as “bad”. To handle that, we reduced the absolute polarity score to 70%. The value of 70% was tuned using the same method as in feature selection parameter tuning (see Section 4.3.2).

To deal with intensifier, a list was compiled for such tokens using a predefined list of intensifiers.[8] To find tokens (root verbs, nouns, adjectives and adverbs), we used similar approach as negation adjustment - the only difference being the score was amplified this time. Amplification factor of 1.5 (50% increase in score) was picked. While conducting experiment, it was found that not all intensifiers increase the score of a root token in Twitter text context. For example, we found the intensifier “too” was used in “this was too good to be true” to reflect sarcasm. In these cases, our intensifier adjustment was further hurting the cause of sentiment classification. Therefore, the list was recompiled that we assumed to rarely being used in sarcastic Tweets. This list can be found in Appendix B.



4.3.4.2 Combining Scores

Once the sentiment (or polarity score) of each token is identified and adjusted, various methods have been used to combine them to build a feature set that can be added in the feature vector. Muhammad et al. (2013) and Yasuhide (2014) and sum up the scores to create one feature value and add the maximum token score as another feature value. Though the authors report this method to improve the performance, the fact that only one value/score was added to the feature vector means that some information may get lost. For example, if there exist one very positive word with score of +6 and five negligibly negative tokens of score +1 each, the final score may sum to 0 indicating that the tweet doesn’t contain

[8] http://en.wikipedia.org/wiki/intensifier

any sentiment. Similar argument can be made about having just a maximal score feature. In an attempt to incorporate more information in the feature vector, we use a novel idea of having six buckets per tweet instance. Each bucket is attributed to polarity and its strength: very positive, positive, mildly positive, mildly negative, negative and very negative. The score ranges of a lexicon is arbitrarily segmented into these 6 categories (see Section 4.3.4.3 below to find exact score boundaries). The count of tokens that belong to each bucket is then added a feature.
We expanded each bucket into four smaller buckets - each belonging to one of the part-of-speech tag in {noun, verb, adverb, adjectives} which were reported by Pang et al. (2002) and Xia et al. (2011) to carry strong polarities. Additionally, this feature set contains more raw information compared to six-bucket feature set above - enabling classification algorithm to have more relevant information at hand. This set of feature was added as an addition to six buckets added initially.

4.3.4.3 Multiple Lexicons

As mentioned earlier, because of the diverse and dynamic nature of Twitter language, a single lexicon fails to capture most tokens that can have high polarity value. In the past researchers have used multiple lexicons to capture tokens from different genres and perspectives (Pang et al., 2002). Our system builds-on this idea to use seven lexicons covering various genres. Some lexicons, like AFINN were compiled manually whereas other lexicons were automatically built using different techniques. Details of each lexicon can be found in Table 3. Some lexicons only included words from standard English language while other more diverse lexicons also included Twitter slang. Additionally, MPQA Lexicon which contained limited tokens, was used merely because it differentiated between same token used in different POS contexts. All of automatic lexicons were compiled by NRC Canada Research Labs and were reported to be successful in various short-form text sentiment classification tasks.

Lexicon Title	Contains Twitter slang	Compilation Method	POS Tags Differentiated	Size
AFINN	Y	Manual	N	2,477
MPQA	N	Manual	Y	8,222
BING	N	Manual	N	6,789
MAX DIFF	Y	Manual	N	1,515
NRC HASHTAG	Y	Automatic	N	54,129
SENTIMENT 140	Y	Automatic	N	62,468
NRC AFFLEX	Y	Automatic	N	45,255

Table 4: Polarity Lexicon Details Used

Each lexicon was given separate independent polarity buckets. This was to avoid merging information that may be incorrect. Separate bucket boundary values were assigned to each lexicon based on how they labeled their data. We noted that automatically generated lexicons contained noise (incorrectly scored tokens). However, the classification algorithm would learn to avoid feature entries assigned to these noisy lexicons (i.e. assign low weights). Since token hits in lexicons increased with an increase in lexicon

numbers, the performance kept increasing which made us decide to keep all the above lexicons in our systems.

4.4 Training Classifier

We built our feature vector using bag-of-words, social media and lexical feature sets as described above. Once the feature vector is built for each instance in the training set, these vectors are then fed to the classifier (learning algorithm) as explained earlier in Section 2.

We used Support Vector Machines (SVM) as our classification algorithm with one-versus-all binary classification type. This method analyzes feature vectors with a labeled class to identify dependency relationship between each feature and a sentiment. To illustrate this process briefly: each vector is considered as datapoint in vector space of dimension equal to feature vector size. Then SVM identifies a hyperplane in this vector dimension that 'best' divides two classes where 'best' is defined as "a good separation is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier."[9] When a new unlabeled tweet is provided to the system, it extracts the feature vector in the same manner as it did for labeled tweets. Finally, the vector is given as an input to the learned model. This function checks on which *side* of the hyperplane (calculated before) the new datapoint lies and a class is assigned. Note that this process would account for two-class classification whereas our problem of sentiment classification contains three classes. To handle that, SVM uses one-versus all binary classification. Using this technique, first it is tested if an instance belonged to one class or not. If the output is true, the process stops. Otherwise, it test for another class whether the instance belong to that class or not. If not, the instance is assigned the third class.

Several considerations were taken into account to make SVM our choice. First, the classifier has been primarily used in research work in the field as discussed in Section 2. Furthermore, Go et al. (2009) also reported SVM performance to be better with bag of words features compared to Naive Bayes and Maximum Entropy. Second, with large feature set due to bag of words, SVM would be able to differentiate irrelevant features from relevant features. Also, feature weights would also help to intensify more relevant features accordingly. As reported earlier, the noise in the dataset is expected to better absorbed by SVM than other algorithms we studied. We used the LibLINEAR implementation of SVM in our experiments (Fan et al., 2008).

4.5 Parameter Tuning

LibLinear comes with several parameters that can be set to different values which in turn can alter the behavior of classifier. We tuned two parameters: C and Class Weights.

4.5.1 C-Parameter

The C parameter tells the SVM optimization how much you want to avoid misclassifying each training example. For large values of C, the optimization will choose a smaller-margin hyperplane if that hyperplane does a better job of getting all the training points classified correctly. Conversely, a very small value of C will

[9] http://stats.stackexchange.com/questions/30166/what-is-the-influence-of-c-in-svms-with-linear-kernel

cause the optimizer to look for a larger-margin separating hyperplane, even if that hyperplane misclassified more points.[10] Therefore, large C values would work best (high accuracy) on instances of training data passed as test set. This often results in overfitting to the training data. In cases where new unseen follows very close patterns as in training data, large C values would be more desirable. However, with Twitter as the dynamic domain, overfitting becomes a bigger problem than usual. Therefore, most previous research works (Pang et al., 2002; Mohammad et al., 2013; Bin Wasi et al., 2014; Yasuhide, 2014) have chosen a small value. Yasuhide (2014), one of the better performing systems lowered the value from default of 1 to 0.03. We conducted standard one-fold-out cross validation parameter tuning to find the most apt value for our system. Values in the range [0.005 to 0.100; equispaced at 0.005] were tried. The method returned 0.01 as C value which was then used in our classifier.

4.5.2 Class Weights: Cost-Sensitive Classification

During initial experimentation, we noticed that our system was performing significantly better in classifying neutral tweets than polar tweets. Upon analysis, it was discovered that this behavior was attributed to unbalanced class distribution in the training dataset. As listed in Table 3, neutral tweets comprised 47% of training dataset whereas negative tweets were merely 15% of the data. As a result, the performance was significantly low for negative tweet classification during experiments. To handle this, the standard concept of cost-sensitive classification was adopted (Batuwita and Palade, 2012). In this type of classification, a varying penalty is assigned to each class. These penalties are communicated to the classifier by assigning weights to each class (defaults to 1 for each class). We assigned weights of 3.3, 2.4, 0.4 to negative, positive and neutral classes. This means that the classifier will be penalised almost eight times more if a negative tweet (in training data) is misclassified than if a neutral tweet is misclassified. Note that during model building phase, we have labeled data and hence we can assign these penalties to different sentiment classes. The weights of each class were picked using standard parameter selection process as discussed earlier. Cost-sensitive classification for Twitter sentiment analysis was also discussed in (Yasuhide, 2014).

[10] http://stats.stackexchange.com/questions/30166/what-is-the-influence-of-c-in-svms-with-linear-kernel

5 Experiments

To gauge the effectiveness of the proposed systems and its features, we conducted experiments on an unseen dataset. The learned model was provided tweets without labeled sentiment class. The model then predicted sentiment classes for these instances that was then compared with the labeled sentiment class. If the predicted sentiment equalled labeled sentiment, it was marked correctly classified instance. For these experiments, Weka, a machine-learning software, was used (Hall et al., 2009).

Together with development and training set, testing dataset was also provided by SemEval 2014. This dataset contained 3,813 labeled tweets following a similar class distribution as the other two provided datasets. To simulate real-world scenarios, we didn't touch this dataset or get inspiration from it throughout our research. Instead, development dataset was used for error-analysis. This is a standard practice to approximate performance of a system in a real-world scenario where tweets would not come with manually-labeled sentiments. The following experiments when conducted after the system development was wrapped up on the aforementioned testing dataset.

5.1 Evaluation Metrics

Two different evaluation metrics were used for our experiments: percent correct (accuracy) and average polar F-score.

For all the experiments, we used accuracy as our evaluation metric. This metric directly reflected the performance on all three classes and gave us a realistic picture of our system's performance in real-life. When picking a metric to identify statistical significance of two models, percent correct was used since it conveyed the performance of the system if it were to be implemented out in the wild. Using these metrics was simpler since Weka (our default platform for this research) provided evaluation on these metrics by default.
We also report Average F-score[11] for polar classes on test set which is the official metric used in SemEval workshops. We used this metric only in our final test set evaluation of our system. The primary reason to use this metric was to compare our system with more than the 70 participants of SemEval three runs of workshop. This was calculated using the following formula:

$$score = \frac{(f1_{neg} + f1_{pos})}{2}$$

SemEval argues that though the score directly includes F1 measure of positive and negative classes, having precision indirectly in the formula also penalizes for incorrectly classified neutral instances.

Our final system achieved the accuracy of 73.53% and average polar F-score of 72.25.

[11] F1 measure (also called F-score) of a given class is the harmonic mean of the precision and recall.

5.2 Impact of Feature Sets

As discussed earlier, feature sets that significantly improved the performance of the system were added to the system. Figure 7 shows this progression on test dataset. Both the aforementioned metrics are reported.

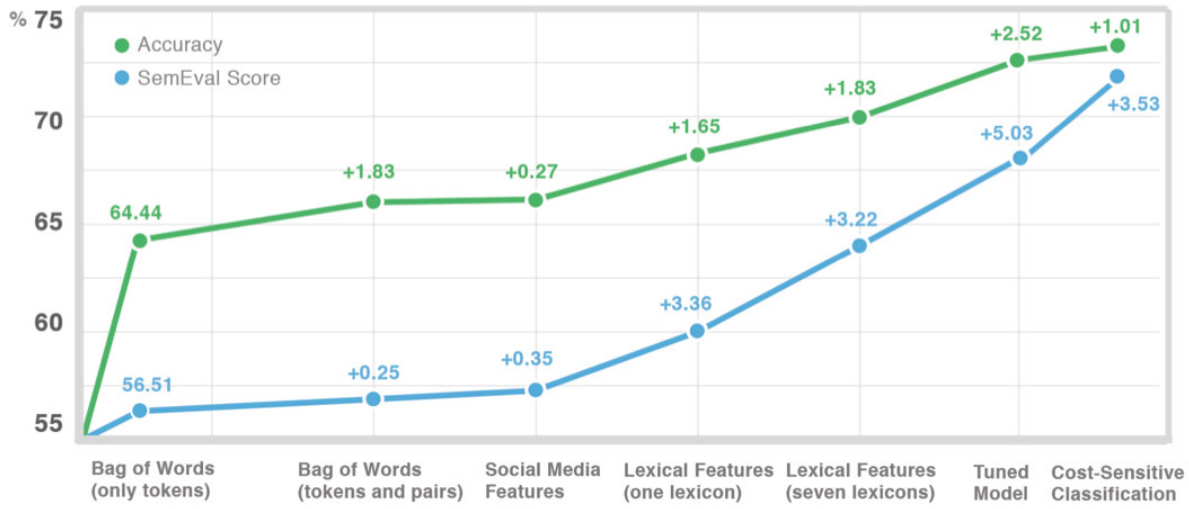


Figure 7: Performance Progression with System Development

Bag of words (only unigrams) was used as a baseline. This is because this feature set has been a standard feature set in the field of sentiment classification. When bi-grams and tri-grams are also added, improvement was noticed verifying that context of the tokens were helpful in sentiment prediction. On the other hand, social media features were not as helpful. As reported earlier, lexical features boosted the performance significantly. The performance was further improved when more than one lexicons were used indicating that adding multiple lexicons give extra coverage over test data and multiple perspectives on token polarities. Setting C value to be low proved helpful too, making the model more adaptive to new unseen data. Finally, cost-sensitive classification significantly boost SemEval score. Since polar classes (which are less in share in datasets) are affected more than the majority class, this technique was expected to increase SemEval score more than the accuracy.

5.3 Comparison

All participants in Semeval workshop reported their average polar F-score performance. This helped us compare our system with other systems. We also compared our current system with the system we submitted in SemEval 2014 shared task. Table 5 summarizes the best scores of SemEval workshops on our testing data. **If our system was to participate in any of these SemEval workshops, it would have outperformed all participating systems in this task.**

System	SemEval Workshop	Rank	SemEval Score
NRC Canada	2013	1	69.02
GU-MLT-LT	2013	2	65.27
Team X	2014	1	72.12
NRC Canada	2014	2	70.75
CMU@Qatar (our last year's system)	2014	17	65.11
Our current system	2015	-	**72.25**

Table 5: Performance comparison on SemEval 2013 Test Dataset

Even though the test data was not seen before final experiments to avoid overfitting To ensure the system performance was not overfitted to just test data of SemEval 2013, we also tested our learned model on two other datasets: SemEval 2014 and SemEval 2015. The performance on these datasets are tabulated below in Table 6.

Dataset	Best Score	Out Score
SemEval 2013 Test	72.12	**72.25**
SemEval 2014 Test	70.96	**71.70**
SemEval 2015 Test	**64.84**	64.34

Table 6: Performance comparison on different test sets

The comparison was limited to SemEval participants -- which included known research labs like IBM Labs, Columbia University, NRC Canada and IIT -- to ensure consistency in test sets and metrics. In order to conduct further comparison with other published works, the testset and metric they reported their results with were required.

6 Interactive Visualization

Our motivation for this research work was to equip brand managers from visualizing and interpreting public sentiments about their brand efficiently. By building a *world-class* sentiment prediction model, we were able to construct an accurate core model that an interface can use for sentiment visualization. However, the task of public opinion visualization was itself challenging and open. This was studied under the context of big-data analysis since the scale of tweets are in millions.

6.1 Background

Our initial research in this area was inspired from Chuang et al. (2012). The work gives guidance as to how data-driven models can be used in visualization for text analysis. The authors argue that for text-analysis to convey meaning to its user, "interpretation and trust" design considerations needs to be accounted for. They argue that visualization should provide trustable meaning to the underlying data patterns. They summarize research work done in the field of text visualization into four design recommendations. First, the data should be visually encoded along appropriate *units of analysis.* Units of analysis should directly respond to the task objectives at hand. Second, the visualization should be verifiable to achieve trust from user's perspective. Third, interactions should be provided to modify visualizations. Fourth, a subset of data should be *progressively disclosed* that was used in building visualization. Any time user finds visualizations out-of-plan (and therefore most helpful in identifying patterns), this data can support the argument made by the visualization. These four guidelines lay the foundation of our research in interactive visualization.

The work of Fisher et al. (2012) motivated us to bring interaction as the pivotal focus. In this work, authors argue that in big-data analytics, interactions play an unmatchable role in giving meaning to the underlying analysis. The work highlights how visuals can reflect the analysis and convert *data into knowledge*. However, this knowledge should be learned without the need of training from developers of the tool. In short, the visualization should be relatable and simple and modes of interaction should be translated from intuitive enquiry of the user.

To identify the gap in the field, several existing public Twitter sentiment analysis tools were studied. [33] models public emotions to analyze socio-economic relationship. Sentiments were encoded onto visuals using in-house computed numerical scores. Though these scores were comprehensive in capturing the sentiment over the dimension of time, we argue that finding meaning in these scores would require training for brand managers.

A popular online tool, Sentiment140.com,[12] captures the sentiment into simple positive-negative sentiment pie. However, it lists tweets used in building the pie chart to gain trust of the user. Also, the real-time nature of the tool makes it useful.

Another publicly available tool is sentiment viz[13] (see Appendix C for a screenshot). The tool's core visualization plots each tweet along two dimensional graph. The first dimension extends from unpleasant to pleasant whereas the second dimension extends from subdued to active. Each tweet instance is assigned a score on each axis and plotted. Though these scores contain information at higher granularity than three-class classification, arguably, these score assignment is much harder task than just assigning one of

[12] http://sentiment140.com

[13] http://www.csc.ncsu.edu/faculty/heale/tweet_viz/tweet_app/

three classes to a tweet. Therefore, we expect the performance of this system to be worse than than of ours. Nonetheless, the tool provided insights about visualizations that were useful in our research.

Based on our research, there was no public tool that followed the four design principles mentioned by Chuang et al. (2012) in the field of Twitter sentiment analysis. We conducted our remaining research to fill this gap.

6.2 Approach

The center theme of all the visualizations was decided to be sentiment analysis. Numerous powerful, effective tools are already published to take advantage of non-sentiment related analysis and visualization. We identified four major areas where targeted visualizations may be effective for brand managers.

- **Time:**
  Analysis of change in sentiment over time was a common theme amongst most tools we studied. Visualization involving time can enable users to identify sudden change in sentiment trends which can lead to pinpointing events that may have led to change in trend. By incorporating interacting to such visualization, the scale of such graph can be adjusted, allowing users to study both long-term and short-term patterns.

- **Geographic Location:**
  Visualization involving maps are also common. Such visuals can help users see the different sentiment distribution over diverse markets. Like before, adding interaction to such visualization can enable user to study sentiment changes over a city and also changes over markets in different continents.

- **User Influencing Power:**
  This was a dimension that is not studied in-depth before. User influencing power is an important metric that businesses are concerned especially on social media. A negative sentiment from highly influencing figure on Twitter can ripple the damage to their followers. Therefore, understanding sentiment distribution along this dimension is essential. With interaction feature, user of the tool can change the visualizations to only focus on tweets by users who have higher influencing power.
  For this project, user influencing power is assumed to be directly correlated to their number of followers.

- **Tweet Platform:**
  This information is relevant to businesses who have different offerings on different platform (i.e. iOS, Android, Web etc.). It can be visualized how sentiments differ based on what platform was used to post the tweet. One use-case for such information can be for mobile app brands that have different application on each platform - possibly offering different user experience.

  The power of sharing feedback and emotions about a brand through Twitter is in its lightning propagation speed. A feedback sent by a user can instantly reach the company. To exploit this power, our objective was to build a real-time analysis tool. As tweets arrive at the system, the visualizations adapt to the newly added information continuously.

6.3 System Design

Our system uses numerous third-party plugins, API and libraries that have boosted development speed manifold. Full information flow and underlying tools used are illustrated in Figure 8.

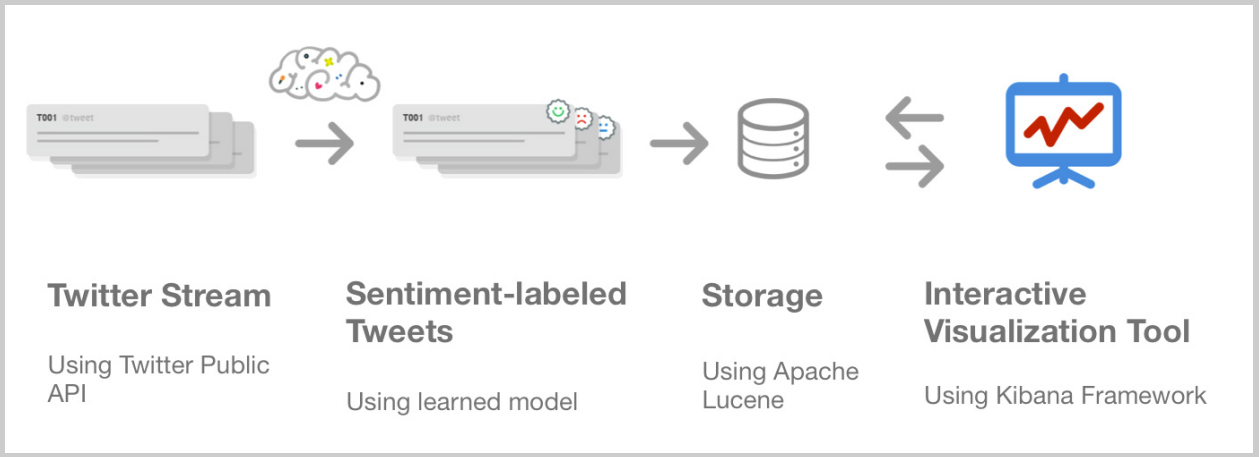


Figure 8: System Information Flow

To fetch tweets live, the Twitter Public Stream API was used. This API required us to subscribe to keywords that enabled Twitter to send a subset of Tweets mentioning that keyword in real-time. This meant that our tool couldn't access tweets that were mentioned in the past and subscription needs to be made for a keyword to capture the trend for an event. We noted that only 4-10% of total tweets mentioning the subscribed trending keywords were received using Public API. To gather higher percentage, the paid Twitter Hose service was required which we avoided. Since we were dealing with big data and had limited financial resources, the size of stream was enough for our experiments.
Along with the Tweet text, meta-data regarding the tweet was also recorded. This included tweeted time, tweeted location, platform used, GPS coordinates and user information who sent the tweet. This user information included their screenname, number of followers, location and account age. It is worth noting that not all fields were provided with each tweet. For example, a very tiny percentage of tweets contained coordinates since the user chose not to attach their coordinates with the tweet.

As soon as the Tweet is fetched by our system, the text is provided to the sentiment analysis learned model that predicted the sentiment it carried. The prediction step was fast enough to not disrupt the speed at which the tweets were received and stored otherwise.

The decision to choose what and where to store was critical for efficiency of the system. Not only we required fast store-speed to not fall-back at incoming stream of data; we also needed to retrieve relevant information at high speed for visualization. Traditional storage techniques like plain text files and relational databases wouldn't achieve real-time analysis that was our requirement. For interactive visualizations, processed metrics couldn't be manually saved due to sheer number of possibilities of filters and options that user can choose from. To solve this problem, the sophisticated information-retrieval architecture of Apache

Lucene was chosen (Bialecki et al., 2012). This architecture is commonly used for internet search engines and logging systems. Instead of saving large text documents to be filtered fast when query arrives, we stored metadata about the tweet and sentiment to achieve fast retrieval at query-time. Also, since the complexity of saving half-baked numbers was handled by Lucene, we could focus on the task of interactive tool without the worry for performance. To communicate to Lucene, interface layer of ElasticSearch[14] is used. This communication uses simple JSON objects that avoided the hassle of learning Lucene interaction language for this project.

To build the web interface for the tool, Kibana framework[15] was used. With limited coding, this framework allowed us to build several kind of visualizations using standard visualization templates. Offered by the same company as ElasticSearch, the framework provided seamless integration with ElasticSearch avoiding the extra effort of developing data models. Also, since the framework is built using AJAX asynchronous web calls, the visualizations can be updated without refreshing the browser. Though, this tool was designed to monitor system logging and status, it performed effectively for Twitter sentiment analysis.

6.4 Outcome

Using the approach and system design described, ten visualizations across multiple dimensions were developed. All these visualizations were laid out on a dashboard as seen in Figure 9.

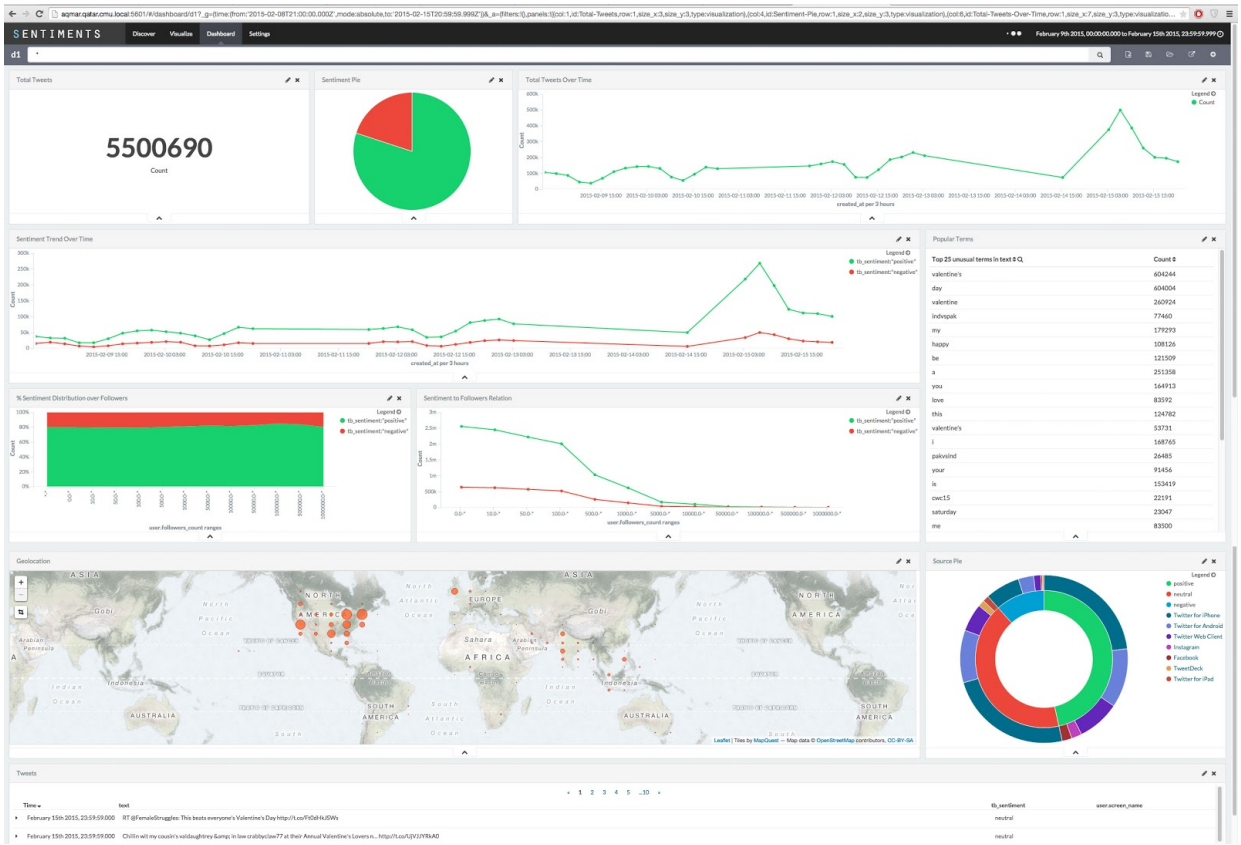


Figure 9: Dashboard of Interactive Visualization Tool

[14] https://www.elastic.co/products/elasticsearch
[15] https://www.elastic.co/products/kibana

6.4.1 Visualizations

6.4.1.1 Plain Metrics:

For any query, we display the count of tweets that match the query. This is referred on the dashboard as total tweets. Also, the change in number of tweets matching the query over time is displayed in a simple-line graph. Each data point on this graph is clickable to filter the query only on the time period represented a single datapoint.

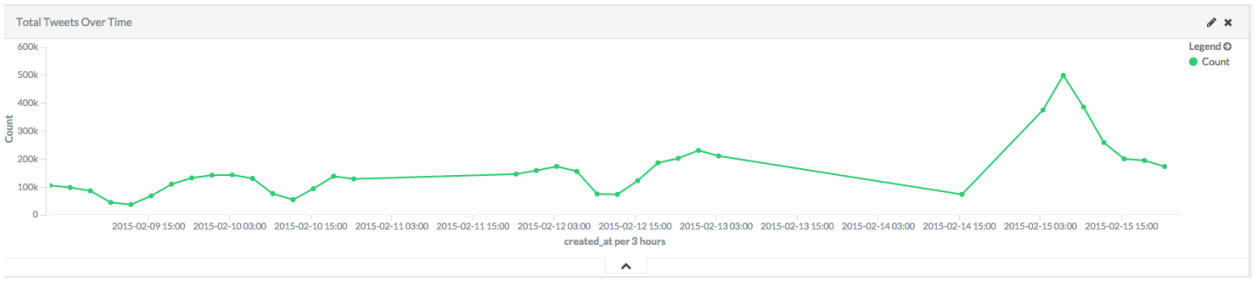


Figure 10: Total Tweets over Time Line-chart

6.4.1.2 Sentiment Pie:

Shows the distribution of positive and negative sentiment over the query. We ignored neutral class tweets here since that was not the focus of this research. Instead, neutral tweets were identified as a 'noise' in sentiment analysis and hence removed. The share of neutral tweets can be identified from Source Pie. When a mouse is overed in any of the pie of this chart, detail information like absolute number of tweets belonging to the pie and percentage shared is shown. Clicking on the pie, would enhance the query to only focus on tweets containing the respecting sentiment.

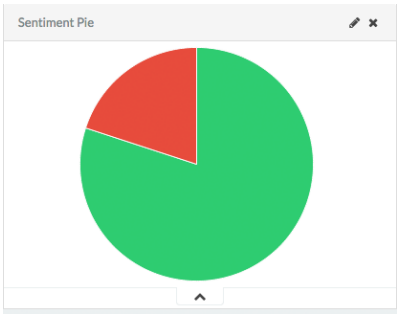


Figure 11: Sentiment Pie

6.4.1.3 Sentiment over Time:

Distribution between positive and negative tweets is analyzed over a period of time. This enables user to identify peaks and possibly identify events causing the peaks. Like in previous line-chart, each datapoint is zoomable to study the representative period closely.

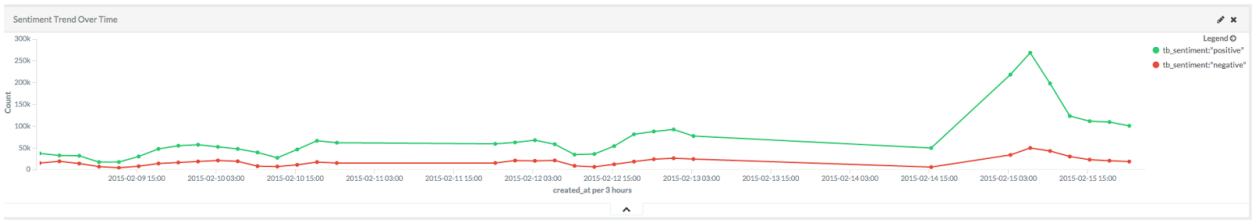


Figure 12: Sentiment over Time

6.4.1.4 Sentiment Distribution over User Influencing Power:

Two visualizations are displayed to reflect the change in sentiment distribution over the change in user influencing power. The first visualization uses an area chart where the y-axis represent the percentage shared between two polar sentiments. This can be seen in Figure 13. This visualization directly communicates how sentiment distribution changes when tweets are restricted to users with higher number of followers. The number of followers are represented on the x-axis at an exponentially increasing intervals. The second visualization displays change in number of polar tweets as the number of followers varies. Like in Sentiment over Time visualization, two separate lines are used to represent positive and negative sentiments. As expected, the lines would be higher towards the left indicating high volume of tweets from users having low number of followers. These lines drop low as they move towards the right indicating low, but impactful, number of tweets sent by users with large following. Clicking on any datapoint would restrict the query to only tweets sent by users having at least given number of followers and carrying a given sentiment.

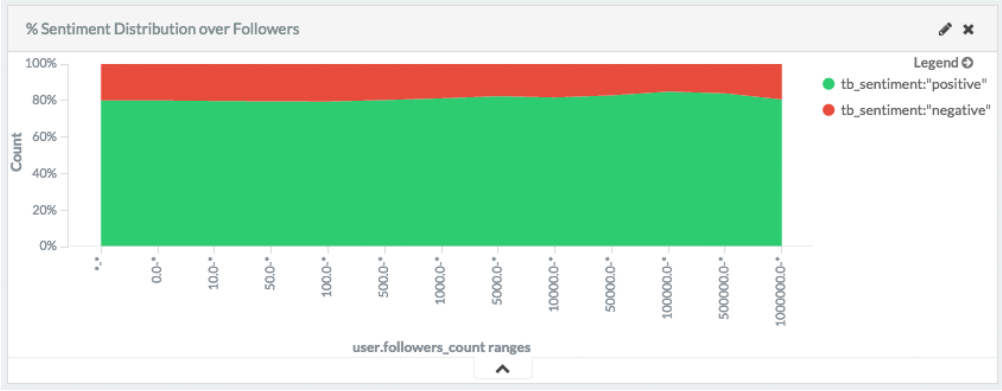


Figure 13: Change in Sentiment Distribution over User Influence

Figure 14: Polar Tweets Quantity over User Influence Line-chart

6.4.1.5 Sentiment Map

Sentiments are overlaid on top of a geographical map that enable user to identify change in sentiment as tweet origin is varied. We found that only a minute share of tweets contains GPS coordinates making this visualization less useful for general purposes. However, we found that certain use cases like "#NepalEarthquake" rendered a higher share of GPS-attached tweets. Tweets responding to the input query was marked on map using a circle - the radius of which reflected the quantity of tweets at the location.

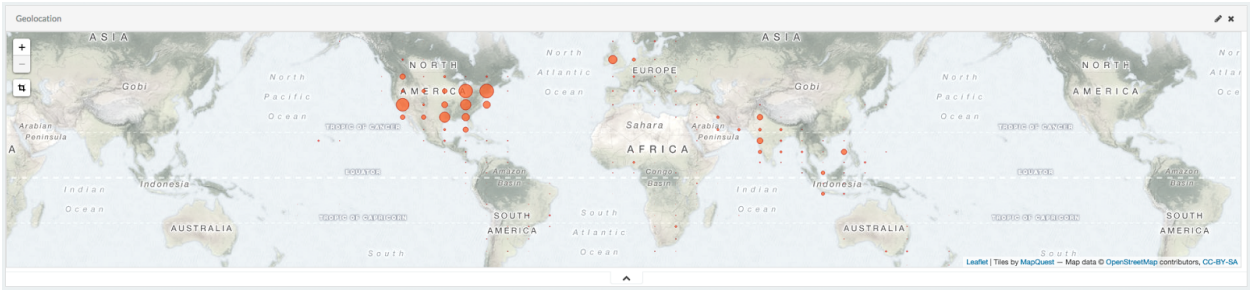


Figure 15: Sentiment Map

6.4.1.6 Source Pie

A two-level donut chart was displayed - showing sentiment distribution at inner level and share of each platform used to post that sentiment on the outer level. This time, since source study is relevant to all three classes, the inner level includes neutral sentiment too. At the other level, the pie is further divided to top 5 sources used to share tweets within the query. Each pie on both levels are clickable. By clicking on the inner level, the query is filtered to the selected sentiment whereas clicking on the outer level leads to further filtering on sentiment and source.

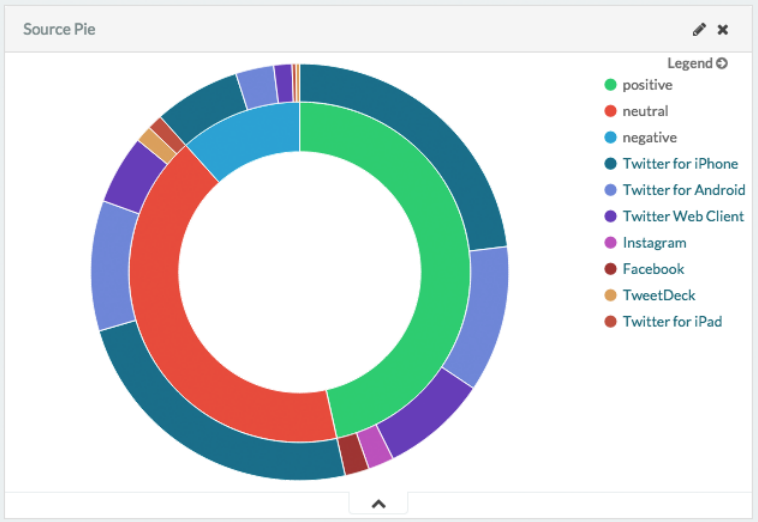


Figure 16: 2-level Source Pie

6.4.1.7 Popular Terms

The tool utilizes text analysis ability of Apache Lucene to extract common words found in the tweets resulting from a query. Using this feature, user can identify common words used in sharing sentiments. Also, the subject of the tweets in the query response can also be discovered. An important use case would be identifying the aspects of a brand that are often discussed in tweets reflecting negative sentiment. Clicking on any of these words update all visualizations to focus on tweets responding to the given query and containing the selected word.

Popular Terms

Top 25 unusual terms in text	Count
valentine's	604244
day	604004
valentine	260924
indvspak	77460
my	179293
happy	108126
be	121509
a	251358
you	164913
love	83592
this	124782
valentine's	53731
i	168765
pakvsind	26485
your	91456
is	153419
cwc15	22191
saturday	23047
me	83500

Figure 17: Popular Terms Occurred in Tweets

6.4.1.8 Raw Tweets

As discussed in Background section, showing a glimpse of underlying data feeding the visualization is essential to gain *trust* of a user. For that purpose, sample 50 tweets are displayed that respond to the given query. The time of the tweet, tweet text, sentiment and user is displayed on a higher level table. However, each row can further expand to show all collected meta-data regarding the tweet.

6.4.2 Real-time

Since Kibana is built on top of AJAX architecture, updating the dashboard with continuous stream of tweets was simple. As the tweets are fetched, the system tests if these tweets needs to be added to the provided query. For example, if the query is limited to negative tweet sent by users having more than 50,000 followers, all tweets that do not match this criteria would not affect any visualizations. On the other hand, tweets that fall under this constraints would be analyzed to produce updated visualizations. Kibana allows this interface update at an interval as low as 5 seconds.

6.4.3 Profiling

Our goal was to achieve effective analysis and visualization of millions of tweets at a reasonable rate. We conducted some informal profiling to measure the rate at which tweets, after getting sentiment attached, can be saved. Also, we conducted profiling over retrieval speed. The system was able to index 1034 tweets each second after analyzing the sentiment on the go. 10 millions tweets were retrieved from the index and visualized by the system in 862 milliseconds.

By conducting sentiment analysis over multiple dimension and building an adaptive dashboard that responds to user interaction, the system achieves interpretation and trust of brand managers and equip them to interpret brand performance real-time - resulting in effective brand monitoring.

7 Conclusions

During this research, we achieved two goals in the areas of machine learning and interactive visualizations.

First, we developed state-of-the-art Twitter sentiment prediction system using supervised text-classification technique and support vector machines. A mixture of proven-published ideas (bag-of-words, lexical features etc.) and novel ideas (polarity buckets, weight-adjusted negation etc.) contributed in achieving a high-performance system. Cost-sensitive classification technique helped in removing bias towards neutral class due to unbalanced training data. On SemEval 2013 test dataset, **we achieved the score (average polar F-measure) of 72.25, higher than any published system**.

Second, we used this sentiment prediction model and Apache Lucene library to build interactive visualization tool empowering brand managers to visualize and interpret public sentiments regarding their brand in real-time. The system was built using design principles guided by Chuang et al. (2012). **This tool visualized sentiments over the dimensions of time, location, platform and influencing power of a user.** The tool could take in search queries to update all the visualizations. These queries can limit the data used for visualization. The adaptive dashboard enabled the system to update visualization every 5 seconds with incoming stream of Tweets.

Several aspects of the system can be improved with further commitment to this research. (1) In building prediction model, an area of ensemble machine learning is trending. Similar tasks have reported performance boost using this technique. (2) Hashtags can significantly help in identifying tweet sentiment. However, since they are compound words, the system was not able to leverage their full potential in indicating sentiment. (3) We reported high performance increase using sentiment lexicons. Since some of these lexicons are automatically compiles and contain some noise, there is a chance of further performance boost if this noise can be reduced. (4) A comparative study of proposed system on diverse short-text datasets would be required in order to utilize this research in other mediums of sharing sentiments online.

The interactive visualisation tool can also be enhanced with further effort. Although using Kibana as an interface framework resulted in quick and simple development, the framework also limited us from experimenting more creative modes of interaction. For example, seven different visualization templates were provided that we could use to construct our own visuals. Modern charts like bubble-chart were not part of this list. Specialized tool built for this task can prove to be more powerful. Additionally, as we noted that only a minute share of tweets contain location coordinates, machine learning can be used to predict these coordinates. Several papers have already been published in this area. Lastly, using free public API, we could only subscribe to limited keywords for live Twitter analysis. It would be more helpful if a system can be built that could use powerful Twitter hose API to retrieve all tweets containing a keyword from past and present. This keyword can also be modified from the visualization tool itself.

8 References

Batuwita, R., & Palade, V. (2012). Class imbalance learning methods for support vector machines.

Białecki, A., Muir, R., & Ingersoll, G. (2012). Apache lucene 4. In *SIGIR 2012 workshop on open source information retrieval* (pp. 17-24).

Bin wasi, S., Neyaz, R., Bouamor, H., Mohit, B. (2014). CMUQ@Qatar: Using Rich Lexical Features for Sentiment Analysis on Twitter. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014).* Association for Computational Linguistics.

Bollen, J., Pepe, A., & Mao, H. (2009). Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena. *arXiv preprint arXiv:0911.1583*.

B. Pang, L. Lee, S. Vaithyanathan (2002). Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 79–86).

Chuang, J., Ramage, D., Manning, C., & Heer, J. (2012). Interpretation and trust: Designing model-driven visualizations for text analysis. In*Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 443-452). ACM.

Councill, I. G., McDonald, R., & Velikovich, L. (2010). What's great and what's not: learning to classify the scope of negation for improved sentiment analysis. In *Proceedings of the workshop on negation and speculation in natural language processing* (pp. 51-59). Association for Computational Linguistics.

Fan, R. E., Chang, K. W., Hsieh, C. J., Wang, X. R., & Lin, C. J. (2008). LIBLINEAR: A library for large linear classification. *The Journal of Machine Learning Research*, *9*, 1871-1874.

Fisher, D., DeLine, R., Czerwinski, M., & Drucker, S. (2012). Interactions with big data analytics. *interactions*, *19*(3), 50-59.

Gimpel, K., Schneider, N., O'Connor, B., Das, D., Mills, D., Eisenstein, J., ... & Smith, N. A. (2011, June). Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2* (pp. 42-47). Association for Computational Linguistics.

Go, A., Bhayani, R., & Huang, L. (2009). Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1-12.

Günther, T., & Furrer, L. (2013). Gu-mlt-lt: Sentiment analysis of short messages using linguistic features and stochastic gradient descent.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter*, *11*(1), 10-18.

Hu, M., & Liu, B. (2004, August). Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 168-177). ACM.

Kavitha, A. S., Kavitha, R., & Viji Gripsy, J. (2012). Empirical Evaluation of Feature Selection Technique in Educational Data Mining. *ARPN Journal of Science and Technology*, *2*(11).

Kiritchenko, S., Zhu, X., & Mohammad, S. M. (2014). Sentiment analysis of short informal texts. *Journal of Artificial Intelligence Research*, 723-762.

Kong, L., Schneider, N., Swayamdipta, S., Bhatia, A., Dyer, C., Smith, N. A. (2014). A Dependency Parser for Tweets. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP).*

Mohammad, S., Kiritchenko S., and Zhu, X. 2013. NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. In *Proceedings of the Seventh international workshop on Semantic Evaluation Exercises*, SemEval-2013, pages 321–327, Atlanta, Georgia, USA.

Nakov, P., Kozareva, Z., Ritter, A., Rosenthal, S., Stoyanov, V., & Wilson, T. (2013). Semeval-2013 task 2: Sentiment analysis in twitter.

Nielsen, F. Å. (2011). A new ANEW: Evaluation of a word list for sentiment analysis in microblogs. *arXiv preprint arXiv:1103.2903*.

Pak, A., & Paroubek, P. (2010, May). Twitter as a Corpus for Sentiment Analysis and Opinion Mining. In *LREC*.

Rosenthal, S., Nakov, P., Ritter, A., & Stoyanov, V. (2014). Semeval-2014 task 9: Sentiment analysis in Twitter. In *Proceedings of the eighth international workshop on Semantic Evaluation Exercises* (SemEval-2014).

Schmid, H. (1995). TreeTagger| a Language Independent Part-of-speech Tagger. *Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart*, *43*, 28.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann (2005). Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis. Proc. of HLT-EMNLP-2005.

Xia, R., Zong, C., Li, S. (2011). Ensemble of Feature Sets and Classification Algorithms for Sentiment Classification. *Information Sciences Journal*, vol. 181, no. 6, pp. 1138-1152. Institute of Automation, Chinese Academy of Sciences (CASIA), Beijing 100190, China

Yasuhide, M. (2014). TeamX: A Sentiment Analyzer with Enhanced Lexicon Mapping and Weighting Scheme for Unbalanced Data. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014).* Association for Computational Linguistics.

Zhu, X., Kiritchenko, S., & Mohammad, S. M. (2014). NRC-Canada-2014: Recent Improvements in the Sentiment Analysis of Tweets. *SemEval 2014*, 443.

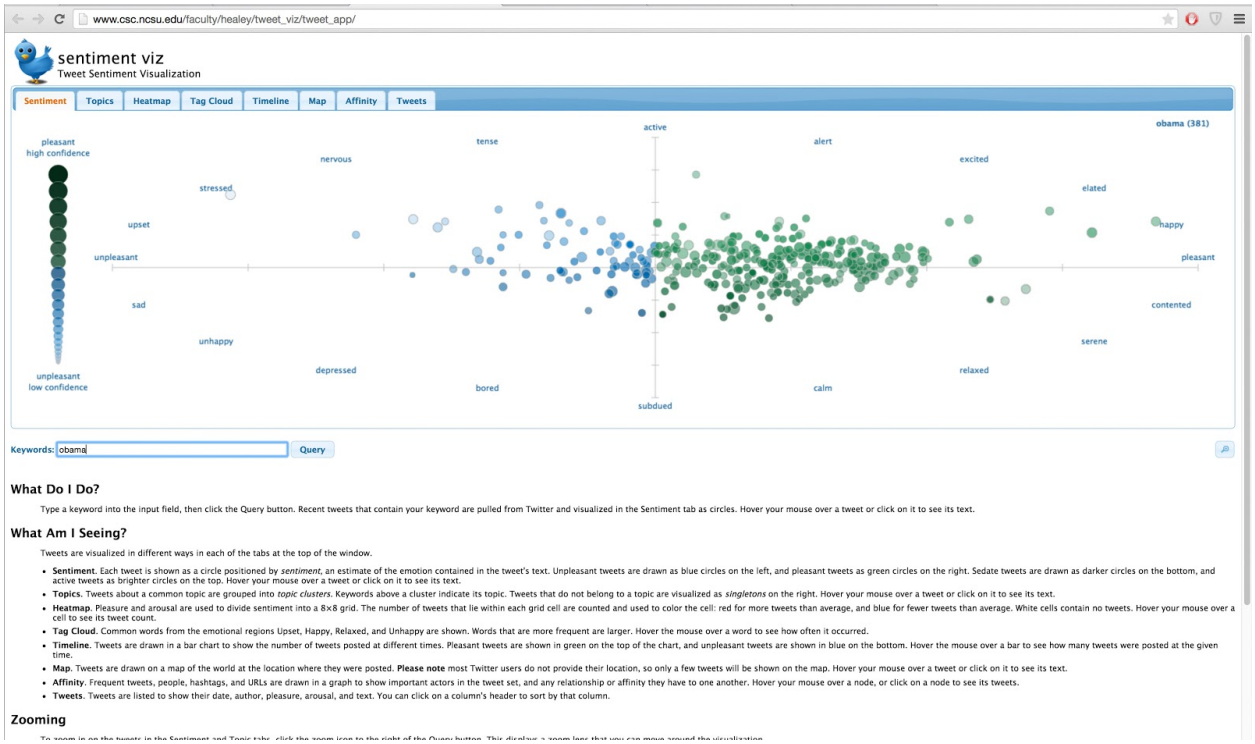
9 Appendix

Negative Emoticons	Positive Emoticons
>:[:-(:(:-c :c :-< :っC :< :-[:[:{ :-|| :@ >:(:'-(:'(>:\ >:/ :-/ :-. :/ :\ =/ =\ :L =L :S :s >.< :| :-| </3 =(	:-) :) :D :o) :] :3 :c) :> =] 8) =) :} :^) :-D 8-D 8D x-D xD X-D XD =-D =D =-3 =3 :-)) :'-) :') :* :^* ;-) ;) >:P :-P :P X-P xp XP :-p :p =p \o/ <3 ♥ ^_^ :* (:

Appendix A: List of Emoticons Used

"most", "fucking", "bloody", "really", "so", "very", "extremely", "super", "surely", "actually"

Appendix B: List of Intensifiers Considered



Appendix C: sentiment viz tool