2017 International Conference on Computational Science and Computational Intelligence

Sentiment Analysis of Tweets Including Emoji Data

Travis LeCompte and Jianhua Chen
Division of Computer Science and Engineering
Louisiana State University
Baton Rouge, LA 70803-4020, USA
e-mail: {tlecom3, cschen}@lsu.edu

Abstract— **Sentiment analysis of text is a valuable tool used to identify and classify bodies of text for various purposes, including public sentiment detection in political campaigns, spam detection and threat assessment. In this paper, we examine the effectiveness of incorporating Emoji data for Twitter data emotion classification. We conducted experiments using Multinomial Naïve Bayes (MNB) and Support Vector Machine (SVM) classification methods, with automatically labeled Twitter data. We compare the accuracy of these classification methods with and without the Emoji data included over varying vocabulary sizes. We find that MNB outperforms SVM on the data for large vocabulary sizes, and both classifiers perform slightly better with the Emoji data included.**

Keywords—Sentiment analysis, Twitter, Emoji, Supervised Learning

I. INTRODUCTION

Natural language processing (NLP) is a field within artificial intelligence that overlaps heavily with linguistics and focuses on understanding data from raw text. It relies heavily on statistical measures to evaluate sources of information and understand text. NLP is employed throughout software today for various purposes, including syntax and spelling evaluation for word processors, human-computer interaction and automatic language translation. With the growing use of the Internet, a largely text based communication media, NLP has grown in interest for security and management solutions, such as spam detection in emails and identification of improper languages in social media postings.

Sentiment analysis, closely related to natural language processing. attempts to identify sentiment (positive, negative, neutral) or emotion from texts and/or social media. It is a challenging task for machines and can even challenge humans occasionally. The structure of the English language, combined with sarcastic tones and ambiguous grammar, can often be misleading for those without an intuitive understanding of the language. Interest in emotion/sentiment detection, however, has grown alongside the field of artificial intelligence, particularly with the increase in emotional communication over the Internet. The advent of social media in the past decade has led many more individuals than ever before to communicate about personal and emotional matters, social and political viewpoints through both private and public messaging platforms. The social media thus has become a rich source of information about public sentiments in many aspects of the society. Developing automated methods for sentiment detection can provide valuable tools for many useful applications, including public sentiment detection for political campaigns such as the US presidential election, and automated detection of abusive communications on social media that is highly desired by social media companies such as Facebook..

II. BACKGROUND

A. Sentiment Analysis

The process of sentiment analysis typically follows a common framework: collect raw data, preprocess the data to remove noise, represent the cleaned textual data in suitable form (typically in a feature vector) for computation, label the data for training purposes, and then feed the data to train a classifier. Preprocessing is necessary due to the ambiguity inherent in language and human tendency to deviate from standard grammatical structures. This results in very noisy data that is difficult to classify. The cleaning process removes some of this noise, resulting in data that is easier to work with. This can involve removing certain punctuation, trimming words of their various endings to result in only root words (stemming), or filtering the data points by keywords or length. Data representation typically takes the form of a vector of features, with the most common feature being the frequency count of a word. This is often called "Bag of Words"(BoW) representation.

Most classification relies on supervised learning, where training is required to teach the system how to recognize the various classes. This requires labeling data with class labels prior to training the system. The labeling process can be done manually by a human to ensure maximum accuracy, though this is slow and subjective. Other methods involve identifying the most prevalent keywords in text and labeling them, though these automatic methods must then be verified by a human to be accurate. If the dataset itself is not labeled accurately, any classification will also be inaccurate.

B. *Classification*

Sentiment analysis of texts essentially amounts to classifying textual documents into sentiment/emotional classes. Various types of classifiers have been used for text classification. The two most commonly used classifiers for text are the Naïve Bayes classifier and the Support Vector Machine (SVM). Both rely on supervised learning, being trained on one dataset with the data points labeled with proper classes and then tested on a disjoint dataset with the class labels withheld. Both classifiers learn the features that define each class and then attempt to find which class fits the unlabeled data of the testing dataset. Space consideration limits

In recent years, use of deep learning neural networks for text classification. In this study we intend to keep the classifiers simple and thus we do not consider using deep learning neural networks here. But we intend to investigate the use of deep networks for sentiment/emotion analysis in the near future.

The Naïve Bayes classifier considered in this study is based on the Bayes' theorem of probability, and the Naïve Bayes Assumption which assumes that the features of an object are conditionally independent, given the class label of the object. There are many variations on the model, such as the Multinomial Naïve Bayes, the Gaussian Naïve Bayes and the Bernoulli Naïve Bayes, all of which operate on the probability of the occurrence of words in the text.

By comparison, SVM is a non-probabilistic classifier. A linear SVM is a binary classifier that aims at finding the maximal margin hyper plane that separates the data points in one class from the data points in the other class such that the "margin" is maximized. This ensures the greatest margin of error for the hyperplane and thus the lowest misclassification rate. See Fig. 1 for illustration.

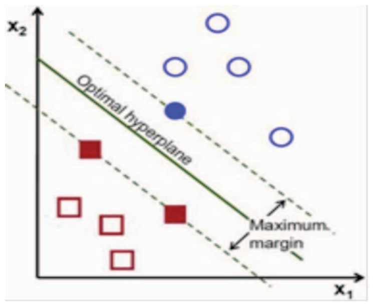


Figure 1: SVM margin representation

There are many variations on SVM, like the variations on Naïve Bayes. Non-linear SVM classifiers can be used to classify datasets that are not linearly separable by using the "kernel trick" with a non-linear kernel function. The idea is to map the data points to higher dimensions and then find the hyper-planes in these higher dimensions where the points may be more easily separable.

C. *Related Works*

Sentiment analysis in general is a very well-studied topic. The process has been applied to text at many granularities, from classifying entire documents [14] to partitioning documents into sections that can be identified by their opinions, allowing for observing how sentiment changes throughout a document [6], or classifying individual phrases by their positivity and negativity [13]. In this way, sentiment analysis can be used for summarizing reviews for products in a comparable manner, detecting whether a given user review is positive or negative for a product. Other work has been done to identify the subjectivity of documents, and whether they should be classified as fact or opinion [5].

In the last decade research has increased involving social media, resulting in many experiments on mining political and subjective opinions from blogs and news articles [4, 8]. There has also been other research into the use of Twitter as a source of data [7, 10, 11, 12, 15], and how sentiment analysis can be executed to accurately identify emotion through these short statements [1, 3]. This is like our work, though we include Emoji data in the tweets that is typically unaddressed in these works.

III. METHODOLOGY

A. *Data Collection*

The first step of sentiment analysis is acquiring a large corpus of high quality textual data for training and testing. The advent of social media in the past decade has brought a new source of highly emotional human text. Many individuals use Facebook, Twitter, or other social media sites every day and post emotional information or opinions. Studies show that upwards of 80% of individuals in the United States with internet access have Facebook accounts, while between 25-30% of these individuals have accounts on various other media websites (Pinterest, Twitter and Instagram for example) [2]. These sites act as an aggregator for this emotional information that researchers can then access and use for experimentation.

For our study, we collected data purely from social media. Specifically, we collected "tweets" from Twitter, which are usually short (less than 140 characters) messages posted by individuals that contain various opinions and emotions. Twitter provides an API for programmers to access their data, requiring a Twitter developer account. We used a package for Python called "twitter" [9] to utilize the API Twitter offers to access their databases and collect the raw tweet data. In this data, all personal identifying information is removed.

To collect specific tweets, we chose to follow the model of Wang et al [11], who tested the viability of tagged tweets as auto-classified textual data. On Twitter, many people use these "hashtags" to mark their tweet as pertaining to a certain topic. For instance, during the 2016 presidential election, many people tagged tweets in favor of candidate Hillary Clinton with "#ImWithHer." This allows for tweets to be grouped and viewed by their tags to quickly access information, and allows for easy classification of tweets. Our tweets were collected from seven broad emotional classes: Sad, Angry, Happy, Scared, Thankful, Surprised, and Love. For each class, we define a set of key words that are indicative of the emotions in that class. For example, for the "Happy" class, the set includes key words such as "joy", "pleased", "glad", "happy", etc. Then in collecting tweets, a tweet is selected if it contains at least one hashtag for a key word from one of the seven emotion classes.

This very human textual data is extremely raw and dirty, without much structure or regulation. Thus, data must be pre-processed for classification. Our collected data was processed to remove retweets, shorter than four words, or tweets containing URLs. This is an effort to purify the dataset and keep only those tweets that are relevant for their emotional content. Overall, we collected approximately 700000 tweets from the 7 classes. After cleaning, around 54000 tweets remained as viable for testing. The distribution of the tweets is shown below. While this seems like an extremely small percentage, this is just the nature of raw textual data. This process discards many potential candidates, and cleans the remaining – removing punctuation, trimming references to people or websites, or deleting non-standard characters.

Table 1 Distribution of Classes After Cleaning

Class	**Number of Viable Tweets**
Sad	23121
Angry	7672
Happy	7710
Scared	4791
Thankful	4525
Surprised	2167
Love	4075
Total	54061

B. *Emoji Data*

The non-standard characters however are of interest for our study due to the recent inclusion of "Emoji" characters in social media. These characters display as certain faces, as seen in Fig 2. Their primary use is to display various emotions such as happiness, anger, sadness and love. Typically the use of these Emoji characters provides very direct and strong suggestions of the emotion included in the text, even though they are commonly thrown away. This is like the case for punctuation, which is often trimmed in many applications except for sentiment analysis, where question marks and exclamation points can provide insight into the emotions of the writer. The purpose of our experimentation here is thus to investigate the influence that including Emoji data within text can have on the accuracy of emotion classifiers.

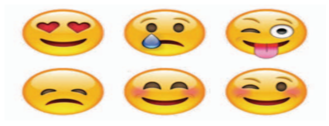
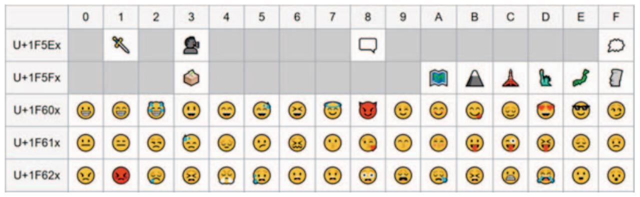


Figure 2: Sample of Emoji characters

These Emoji characters are represented in the text as various Unicode values. Web browsers then map these Unicode sequences into the associated images to display. The same process can also be used when collecting the purely textual data – collect the Unicode values and map them to the correct Emoji faces. An example of the Emoji encoding scheme is shown below in Fig 3. For example, the Unicode sequence U+1F600 results in a smiling face, while U+1F622 results in a crying face. Instead of mapping the Unicode back into an image, since our plan is to work solely with textual data, we can map the Unicode sequences into specific textual sequences, such as <ANGRYFACE> for a mad face and so forth. This allows us to capture the Emoji data in a human recognizable form in text.



0 1 2 3 4 5 6 7 8 9 A B C D E F
U+1F5Ex
U+1F5Fx
U+1F60x
U+1F61x
U+1F62x

Fig. 3. Example Unicode Encoding Scheme for Emoji

C. *Data Labeling*

Recall that our sentiment analysis task is to classify tweets into one of the 7 seven broad classes – sad, angry, happy, scared, thankful, surprised, and love, following a similar model to Wang et al [11]. In collecting the tweets, we searched for tweets that included one of the "hashtags" that are closely related to these seven emotional categories. After collecting the raw data and cleaning it, the resulting data must then be labeled with classes to allow for training and testing the classifier. Typically, one would have to label each tweet by hand to ensure correct labeling. However, exploiting the presence of hashtags in these tweets, an automated labeling process is used (per Wang et al [11]) to label tweets using the hashtags. For each tweet, the method would first search the tweet for hashtags that belongs to one of the 7 groups corresponding to the emotion classes. If there are more than one such hashtags, locate the last one in the tweet and label the tweet by the emotion class associated with that hashtag. In addition the hashtag is removed from the tweet. As shown in Janssens et al [3], the hashtags can be used for automatically classifying the emotion of the associated tweets with high accuracy. For example, a tweet containing the text "#happy" can be assumed to have a happy message.

To ensure the accuracy of the automated tweet labeling process, we need to verify that the automatic labeling of tweets via their hashtags accurately represents what a human would gather from the tweets. To achieve this, we randomly sampled 100 tweets from the dataset and had two human annotator label the tweets independently by hand. Comparing the manual classification labels to the labels generated by automated labeling process using hashtags, we achieved an agreement rate of about 70% for each annotator. Therefore, we believe the automatic labeling does a reasonable job of approximating the true classification of each tweet.

D. *Representation*

At this point the data is cleaned to contain only relevant information and labeled for the correct emotional classes. To continue with the classification process, the tweets must be transformed into some numerical representation to feed to a classifier. The most common representations focus on the frequency of the words or characters present in the text. For our experimentation, we utilize both the standard bag of words representation and a bigram representation, both of which rely on word frequency.

E. Experiments

For each of the tweets that pass the previously described cleaning and labeling process, we represent them using both a bag of words and bigram approach. For the bag of words approach, we counted the occurrence of every word present in the dataset. This resulted in a total vocabulary of approximately 10000 words. To reduce computation time, we then restricted the vocabulary to subsets of only the most frequent words. The subset sizes we used were 100, 200, 500, 1000, 2500, and 5000 top words.

For the bigram approach, we again relied on only the most frequent 1000 words. However, for each tweet, instead of measuring the frequency of each individual word, we measure the frequency of each pair of words from the top 1000 words. Overall this results in 1 million possible bigrams (1000^2) to consider. To reduce computation time, we again restricted the set to only the top 10000 and 20000 bigrams, rather than consider all 1 million. Therefore, every tweet is represented as a vector of length 10000 (or 20000) that can be fed to the classifier.

Within each approach, we considered the same tweet dataset in two versions, one including the Emoji data in the tweets, and the other with the Emoji data cleaned out. Our goal is to compare both versions of the dataset with both the bag of words and bigram approach on both the Naïve Bayes and SVM classifier. For the purposes of testing and training, we have varied the training/testing split to 30/70, 50/50 and 70/30. All trials have been run on the same machine under the same conditions to provide consistency between the various trials.

IV. Results and discussions

For each of the experimental cases, we measured the classification accuracy of the system using the given vector size by running each trial ten times and averaging the accuracy to remove variation.

Table 2. Classification Accuracy of trials (30/70 split)

30/70 split	**1%**	**2%**	**5%**	**10%**	**25%**	**50%**
BOW NE MNB	0.5289	0.5516	0.5775	0.5963	0.6151	0.6251
BOW R MNB	0.5297	0.5496	0.5778	0.5981	0.6211	0.6302
BOW NE SVM	0.54104	0.56486	0.5917	0.5964	0.58244	0.57886
BOW R SVM	0.5399	0.5649	0.5919	0.5992	0.5882	0.58492
BG NE MNB	0.56446	0.573	NA	NA	NA	NA
BG R MNB	0.56414	0.5708	NA	NA	NA	NA

In Table 2, the values in the cells are the classification accuracy over test data. Note that we used 30 percent data for training and 70 percent data for testing. The trials are labeled to describe their design (BOW = bag of words, BG = bigram, NE = no emoji, R = emoji, MNB = Multinomial Naïve Bayes, SVM = Support Vector Machine), and the size columns denote the size of the representation vectors (for bag of words it is the % of 10000, bigram the % of one million). The higher percentages for bigram have not been computed as the computation time became too excessive.

For the sizes that have been calculated for bigram, the bigram representation outperforms the bag of words representation. This is expected as the bigram retains contextual information. However, the accuracy increase from a vocabulary size of 100 to 200 words is smaller for bigram than bag of words, and it seems possible that the bigram would fall below the bag of words representation rather than retaining greater accuracy rates. This is not entirely surprising, as the short length of tweets means many bigrams are not present, resulting in very sparse vectors. These sparse vectors in turn are very similar to each other and can be difficult to classify.

The results for the bag of words representation are shown in Fig 4. Comparing the two classifiers, SVM outperforms MNB until a vocabulary of 1000, at which point MNB becomes more accurate. Both classifiers however follow a similar linear increase in accuracy up until a vocabulary size of 500 words. At this point, SVM levels off and proceeds to become less accurate, while MNB continues becoming more accurate until beginning to plateau at a vocabulary size of 2500. This is possibly due to the difficulty for SVM in finding the hyperplane at such a high dimension, while the MNB algorithm is not as greatly affected by an increase in feature descriptor size.

Within each classifier, we must also compare the effect of the Emoji data on the accuracy. Neither classifier shows much difference in accuracy until a vocabulary size of 1000. At this point, trials including the Emoji data begin to become more accurate than their counterparts without the included Emoji data. While this is what we hoped for, the margin is very small, with possibly approximately a 0.5% difference in accuracy or less. We believe this is due to the Emoji data largely agreeing with the included text, and thus not providing much more insight than the text itself.

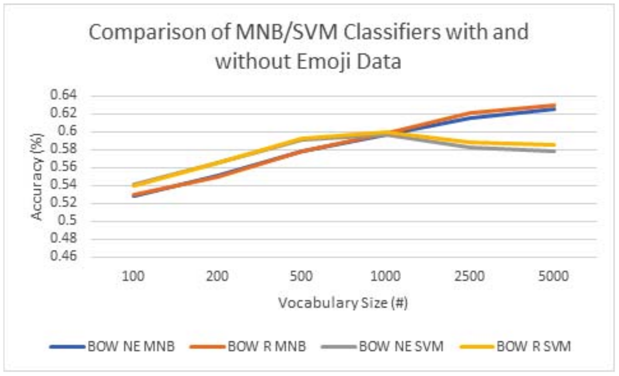


Fig. 4. Accuracy Results with and without Emoji

The results are similar when varying the training/testing split sizes. Though the absolute accuracy rates generally increase, the general accuracy pattern is the same. The tables of the 50/50 split and 70/30 split are each shown next.

Table 3. Classification Accuracy (50/50 split)

50/50 Split	**100**	**200**	**500**	**1000**	**2500**	**5000**
BOW NE MNB	0.52923	0.55543	0.57864	0.60159	0.62349	0.63074
BOW R MNB	0.52999	0.55357	0.58357	0.60499	0.63125	0.63785
BOW NE SVM	0.54126	0.56861	0.59669	0.61577	0.60822	0.59339
BOW R SVM	0.53959	0.56544	0.59847	0.61523	0.60879	0.59505

Table 4. Classification Accuracy (70/30 Split)

70/30 Split	**100**	**200**	**500**	**1000**	**2500**	**5000**
BOW NE MNB	0.53311	0.55142	0.58325	0.60872	0.6280	0.63583
BOW R MNB	0.53173	0.55733	0.58459	0.6106	0.6325	0.64275
BOW NE SVM	0.54026	0.56657	0.60181	0.62309	0.6253	0.61084
BOW R SVM	0.54211	0.56582	0.60237	0.62553	0.6279	0.60609

Each of the classifiers displays approximately a 1% accuracy increase in the 70/30 split over the performance of the 30/70 split, while retaining the same general shape. Again, the best performing classifier is the MNB classifier with Emoji data included. To better compare across the training and testing sizes, we selected this classifier to compare across the varying sizes. This is shown in the graph below (Fig. 5).

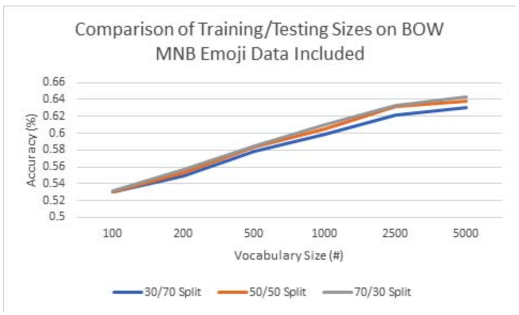


Fig. 5 Accuracy of MNB with Emoji

There is an accuracy increase by increasing the training data set size. However, the increase magnitude is relatively low, and changing the size does not greatly affect the shape of the graphs whatsoever. They each become more accurate as the vocabulary size is increased, and the training size change has greater affect at higher vocabulary sizes. Due to the continued lack of improvement with the Emoji data in various trials, we created a confusion matrix for the best classifier recorded (BOW MNB emoji data included, 30/70 split with 5000 vocabulary size). This matrix maps the actual classes of tweets to the classes predicted by the classifier. The diagonal of the matrix displays the correct classification rates, while all non-diagonal entries display various misclassification rates.

Table 5. Confusion Matrix

	Sad	**Angry**	**Happy**	**Scared**	**Thankful**	**Surprised**	**Love**
Sad	**0.86557**	0.03346	0.03847	0.03081	0.01695165	0.00117925	0.01356
Angry	**0.60518999**	**0.27248**	0.03985	0.04727	0.02502317	0.00092678	0.00927
Happy	0.24679345	0.02565	**0.55551**	0.02963	0.09066785	0.00044228	0.05131
Scared	**0.60244898**	0.02531	0.06776	**0.26776**	0.02448980	0.00081633	0.01143
Thankful	0.30249480	0.01247	0.16840	0.02286	**0.46777549**	0	0.02599
Surprised	**0.81081081**	0.05405	0.06757	0.02703	0.02702703	**0**	0.01351
Love	0.29465302	0.01707	0.24232	0.02844	0.08304891	0	**0.33447**

The diagonal elements (correct classifications) have been bolded in the confusion matrix, similarly for any occurrence of misclassification greater than correct classification. Most of the correct classification rates are moderately successful – sad, happy and thankful are very commonly classified correctly. Sad dominates the classification accuracy due to its massive size relative to the other classes. Love is commonly misclassified as either sad or happy, though it is still correctly classified as the most common outcome.

The large misclassification rates are prevalent in the remaining classes – angry, scared, and surprised. These are misclassified as sad more often than they are classified correctly. The worst case is surprised, being misclassified as sad over 80% of the time. This is likely due to the relatively small size of surprised to the large size of sad, which also explains why the other classes are almost never misclassified as surprised. Angry and scared perform better, though they are still misclassified more often as sad than their correct classes.

Overall it seems that the classifier struggles to classify correctly due to the relative sizes of the classes. This also affects the accuracy rates of the Emoji data, which are not as prevalent in our experiments due to issues in the data sampling. Since some of the emotion classes show up much more frequently than others (happy and sad are much more prevalent than thankful) which skews the classification process towards predicting sad more than any other class. Additionally, our data was collected throughout the 2016 United States presidential election, a social phenomenon that resulted in many individuals taking to Twitter to express very positive and negative emotions regarding the event. We believe this additionally skewed the dataset, making prediction more difficult. For instance, candidate Donald Trump has a mannerism of calling things "sad," which led to many people

tweeting with the hashtag "#sad" while discussing things not related to the emotion, leading to much confusion.

V. CONCLUSIONS

Our experiments have shown the effects of including Emoji data in text for sentiment analysis. Although minor, there is an improvement in classification accuracy for including the Emoji data without any substantial computational overhead. We additionally compared both MNB and SVM classifiers on the data, with both bag of words and bigram representations. SVM outperforms MNB at small vocabulary sizes, though this is opposite at large vocabulary sizes. Bag of words also seems to outperform the more advanced bigram approach while saving computation time, as the bigram vectors are very sparse due to the short tweets. Overall this suggests that one should use a bag of words representation with an MNB classifier for large vocabulary sizes including the Emoji data to achieve maximum classification accuracy.

We plan to perform more experimentation on varying the training/testing ratio to produce learning curves for both the MNB and SVM implementations. Additionally, we want to experiment with other possible representation formats apart from bag of words and bigram. Lastly, we would advise ensuring more evenly distributed data for training and testing if possible, as this can skew the classification process and lead to high misclassification rates.

REFERENCES

[1] Agarwal, Apoorv, et al. "Sentiment analysis of twitter data." *Proceedings of the workshop on languages in social media*. Association for Computational Linguistics, 2011.

[2] Greenwood, S., Perrin, A., Duggan, M, "Social Media Update 2016," *PEW Research Center,* 11 Sept 2016. Web. Accessed April 2017. http://www.pewinternet.org/2016/11/11/social-media-update-2016/

[3] Janssens, O. et al. "Real-time Emotion Classification of Tweets," *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 2013.

[4] Liu, B. "Sentiment Analysis and Opinion Mining," *Synthesis Lectures on Human Language Technology,"* May 2012.

[5] Liu, Bing. "Sentiment analysis and subjectivity." *Handbook of Natural Language Processing, Second Edition*. Chapman and Hall/CRC, 2010. 627-666.

[6] Nasukawa, T. and Yi, J. "Sentiment analysis: capturing favorability using natural language processing," *K-CAP Proceedings on the 2nd International Conference on Knowledge Capture*, 23 Oct 2003.

[7] Pak, A. and Paroubek, P. "Twitter as a Corpus for Sentiment Analysis and Opinion Mining," LREc. Vol. 10. No. 2010. 2010.

[8] Pang, B. and Lee, L. "Opinion Mining and Sentiment Analysis," *Foundations and Trends in Information Retrieval*, 7 July 2008.

[9] Python Twitter Package. Accessed 5 April 2017. https://github.com/bear/python-twitter

[10] Saif, H. et al. "Contextual semantics for sentiment analysis of Twitter." *Information Processing & Management*, Volume 52, Issue 1, 2016, pp 5-19.

[11] Wang, W. et al. "Harnessing Twitter 'Big Data' for Automatic Emotion Identification" *ASE/IEEE International Conference on Social Computing,* 2012.

[12] Wehrmann, J. et al. "A character-based convolutional neural network for language-agnostic Twitter sentiment analysis." *2017 International Joint conference on Neural Networks (IJCNN)*. Anchorage, AK, 2017, pp. 2384-2391.

[13] Wilson, T., Weibe, J. and Hoffman, P. "Recognizing contextual polarity in phrase-level sentiment analysis", *HLT Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, 6 Oct 2005.

[14] Zhang, C. et al. "Sentiment analysis of Chinese documents: From sentence to document level," *Journal of the American Society for Information Science and Technology.* 2 Sept 2009.

[15] Zimbra, D., Ghiassi, M. and Lee, S. "Brand-Related Twitter Sentiment Analysis Using Featur3e Engineering and the Dynamic Architecture for Artificial Neural Networks." *2016 49th Hawaii International Conference on System Sciences (HICSS)*, Koloa, HI, 2016, pp. 2930-2938.