Chapter 1

OPTIMIZING SHORT TEXT SENTIMENT ANALYSIS FOR MOBILE DEVICE FORENSICS

Oluwapelumi Aboluwarin, Panagiotis Andriotis, Atsuhiro Takasu and Theo Tryfonas

Abstract In recent times, mobile devices have served as a dominant medium for communication across individuals. While communicating, humans express different emotions which can be analyzed to deduce their emotional inclination on diverse subjects of interest. Natural Language Processing (NLP) techniques have been used to analyze sentiment in text by researchers. However, most research work involving sentiment analysis in the short message domain (Short Message Service texts: SMS and Twitter feeds: tweets) do not account for the presence of non-dictionary words. In this paper, we investigate the problem of Sentiment Analysis in short messages and analyze the emotional swing of an individual over time. This provides an additional layer of information for forensic analysts while probing suspects. The Maximum Entropy algorithm was used to classify short messages as positive, negative or neutral. Non-dictionary words were normalized and the impact of normalization and other features on the classification task was evaluated. The classification F-Score was improved when compared to our previous work and approximated the current state-of-the-art system's performance. We also developed an intuitive user interface to ease extraction of sentiment information from SMS. This user interface presents a useful starting point for forensic analysis when trying to identify points of emotional variation in short messages. The timeline view within the forensic tool offers the capability to a forensic analyst to identify periods that require further investigation. Finally, we used Apache Solr for indexing to ensure the analyst retrieves desired information fast and efficiently using facetted search queries.

Keywords: Sentiment Analysis, SMS, Twitter, Normalization, Text Mining, Microblogging.

1. Introduction

The ubiquity of mobile devices has extensively redefined the communication landscape across the world. This has led to the creation of valuable individual data through conversational services like SMS and micro blogging platforms like Twitter. Mining the content of such interactions can provide valuable insight on the people an individual communicates with. Information such as the time the interaction takes place and the content of the conversation might be useful to the forensic analysis because it reveals patterns hidden in the text.

Using Machine Learning techniques, additional information on the disposition of conversations can be extracted. Sentiment Analysis is the field concerned with retrieving opinion or emotion expressed in text. In literature, applications of Sentiment Analysis have been proposed in different fields with a special interest in the social media and micro blogging services. Sentiment information can also be useful for forensic investigations in smartphones as proposed by [3] and [4]. This paper investigates the use of sentiment analysis to model the emotional swing of an individual as opposed to the emotional swing of a group of people towards a brand, which is more common in literature. We used Machine Learning algorithms for the sentiment polarity classification task. We also accounted for lexically incorrect terms that are prevalent in conversational texts by normalizing them. Such invalid terms are known to negatively impact the efficiency of NLP tasks as proposed by [23]. The emotional timeline generated by the tool we developed provides an additional layer of information about a person under investigation, because it helps the forensic analyst identify periods of time that such an individual exhibits volatile emotional state. Our main contributions are outlined below:

- We show the effect of normalizing non-dictionary words alongside other sentence level features to improve the Sentiment Polarity classification task. A ‘Part of Speech’-Tagger, (POS-Tagger) aware of the peculiarities on short messages was also shown to improve the classifier performance.

- We study how individual features affect the performance of the most efficient classifier for emotional classification in short text messages (SMS).

- We present an extended version of our conceptual design of a forensic tool that provides details about sentiment polarity expressed in an individual’s SMS messages in a concise and intuitive manner to facilitate rapid extraction of information by forensic analysts [1].

2. Related Work

The need to know the opinion of others on subjects of interest always proves valuable when trying to make decisions in an unfamiliar terrain [7, 20, 24, 33]. The ubiquity of reviews and recommendation data provided online, makes the web a go-to place for seekers of such information. People rely on the opinion of others on the web to guide decision-making by viewing reviews of products, movies, employers, schools and so on. Increased interest in this sort of information has been the major driver of research into the field of Sentiment Analysis. Sentiment Analysis started getting increasing attention in the research landscape after the work presented at [25] and [33] in 2002 and since then it has been studied extensively, leading to its use in many interesting applications like content advertising [16], election monitoring [34] and customer feedback data classification [12].

Sentiment Analysis problems often take the form; given an instance of text, determine its polarity as either positive or negative, or identify its position within the extremes of both polarities [24]. Since some text instances are neither positive nor negative, sentiment analysis also involves identifying texts that do not convey any form of emotion which are referred to as 'neutral'. Hence, sentiment analysis problems are handled as classification or regression tasks. As an example, deducing if a movie review is positive or negative is a binary classification task, while deducing how positive the review is on a scale of 1-10 is a regression task. In addition, such problems can be treated as multi-class classification tasks in scenarios where the instances to be classified fall under categories such as positive, negative and neutral.

Sentiment Analysis techniques include: a) Lexicon based methods [3], b) Machine Learning (ML) methods [4] and c) Hybrid approaches combining both methods [4, 11]. When treating sentiment analysis as a classification task, ML algorithms known to perform well in text classification are often used. Some of the supervised learning algorithms commonly used in literature are; Support Vector Machines (SVM), Multinomial Naive Bayes (MNB) and Maximum Entropy (Logistic Regression) [25, 13, 32].

In Digital Forensics, text mining methods have been used in tasks like authorship attribution in emails [17] and text string searching [5]. SVM algorithms were also used to determine the authors of emails [10] or identify the gender of the author of an SMS [9]. Authorship attribution experiments were also conducted using ML in [15] and [30]. The work of [21] is closely related to our research as it focuses on extracting sentiment polarity information from Twitter feeds (tweets). The paper details the techniques used by the team that reached the highest accuracy (F-score) at the 'SemEval-2013 Task 2: Sentiment Analysis in Twitter' competition [22] for Sentiment Polarity classification.

Additionally, in [3] we initiated a research that involves the use of sentiment analysis to augment digital forensic investigations by retrieving opinion information from SMS found on mobile devices. A lexicon-based technique was used for the Sentiment Polarity Classification task and a proof of concept was presented to visualize mood patterns extracted from SMS databases. The maximum classification accuracy we achieved was estimated to be 68.8% (for positive SMS messages). The classification F-scores were improved in [4] and for binary classification (SMS: positive superclass and negative class) we measured that F-score was 74.4%. However, in this work we included neutral and positive messages in a superclass and this fact resulted to large False Positive Rates (FPR). These rates were decreased drastically with our hybrid classifier [4] but the total estimated F-score also decreased (62%) when we performed a three-class categorization (positive, neutral, negative). The best sentiment classification performance (for SMS) so far in the literature was achieved in 'SemEval-2014 Task 9: Sentiment Analysis in Twitter' contest [27]; the team that performed better, reached an accuracy that was estimated with F-score = 70.28% for classifying SMS in three classes. However, there is no information about the FPR this classifier produces.

Since ML techniques are known to outperform lexicon-based methods [13], we decided to use ML methods for sentiment classification in order to enhance the accuracy of our forensic tool. Our work takes cues from some sentence level features presented in [21]. We further improved the classification outcome by integrating normalization of non-dictionary words. We also used conclusions presented at [31], which is a survey that describes commonly used techniques in handling noisy text and serves as a good introduction to the problem we investigate. Finally, the Statistical Machine Translation (SMT) technique for normalization (presented in [14]) served as the basis of our normalization task in the current work.

3. Dataset and Classification

We trained our classifier using the Multinomial Logistic Regression algorithm also known as Maximum Entropy (ME). The ME algorithm makes it possible to apply Logistic Regression to multi-class classification problems like the 3-class short message classification task that is handled in this work. ME is usually preferred to the Multinomial Naive Bayes (MNB) algorithm because it does not assume statistical independence of features as the MNB does. It therefore implicitly takes NLP properties (like negation) into consideration when creating models. In terms of training time, while the ME is relatively slower than MNB, its training time is significantly lower compared to algorithms like the SVM which was used in [21]. The algorithm was implemented in Python using the scikit-learn library [26]. Parameter tuning was carried out by a process called 'Grid Search' in scikit-learn, which involves specifying a range of parameters and allowing the system run through different permutations to identify the optimal combination. Subsequent parts of this section describe an overview of the dataset, features used and pre-processing techniques.

3.1 Dataset Overview

The dataset used during the SemEval-2013 competition was utilized for training our models [28]. Our training dataset contained a total of 8,120 tweets (positive: 37.2%, negative: 14.7% and neutral: 48.1%). Furthermore, we used the testing dataset from our previous work [2] making it possible to compare results directly.

3.2 Pre-processing

Pre-processing involves the cleaning of raw datasets before applying ML algorithms. It is a standard procedure in most ML tasks and the techniques used vary across domains. Proper pre-processing ensures noisy data is in proper shape for ML algorithms. In text mining, pre-processing often consists of normalization, spelling correction, managing text encoding, etc. Some of the techniques used here are briefly described subsequently.

Normalization. In this context, normalization involves resolving lexically incorrect monosyllabic terms to their correct form. This may be in form of spelling mistakes or ad-hoc social media short forms as defined by [19]. Normalization is known to improve the quality of some NL tasks like language translation as seen in [18] and [19]. Normalization was handled as a Statistical Machine Translation (SMT) task and some of the techniques used are described in [19]. The outcome of the normalization task is a dictionary mapping of lexically incorrect terms to the lexically correct variants. An example of such mapping can be identified by mapping each word in the 'Raw text' to the corresponding word in the 'Normalized text' in the following representation.

Raw text: Hi ranger hw r u

Normalized text: Hi ranger how are you

The SMT technique requires the use of a parallel corpus. A parallel corpus is a list of messages containing lexically incorrect terms mapped to the lexically correct form. In our dataset the total number of 'incorrect terms' that were mapped as 'corrected terms' using the aforementioned method was 156. Thus, the generated normalization dictionary was quite small due to the limited corpora size. To supplement this disadvantage, we used as an addition the normalization dictionary generated from the work of [14] which contains over 41,181 normalization candidates in the short message domain.

To apply the normalization dictionary to the corpus, each tweet was tokenized and lexically correct tokens were filtered off, leaving only lexically invalid tokens. Lexically correct terms were identified based on their presence in an English dictionary using the Python 'Enchant' library. Enchant is a python spell checking library. It can be used to identify words that are not in the dictionary of a defined language. We identified the remaining lexically correct terms by checking for their presence in online slang dictionaries (Urban Dictionary). Normalizing the data instances before the sentiment polarity classification task is one of the main contributions of this work.

Data Cleaning. Some terms (specific to Twitter and SMS) were cleaned to reduce the noise in the data. All occurrences of a user mention (e.g. @jack) and all web addresses within tweets were replaced with an empty string. In addition, occurrences of the term 'RT', which means retweet on Twitter, were removed. These terms were removed to avoid over-fitting the model on the Twitter dataset knowing that mentions, retweets and URLs are not as common in SMS as they are in tweets. Positive emoticons were replaced with words known to have a positive connotation while negative emoticons were replaced with negative polarity words. This ensures that the information emoticons add to the model is not lost during the tokenization process, since emoticons are prone to ambiguous tokenization.

Data cleaning also involved the unification of elongated expressions. Elongated expressions here are terms with a sequence of three or more characters (e.g. 'whyyyy'). Such expressions are commonly used to convey emphasis on social media and the number of elongated characters often varies across users. All such elongated characters were trimmed to a maximum of two, making it easier to identify words that are emphasizing the same emotion. This implies that a term like 'killll' was trimmed to 'kill'.

Stemming. This is the process of reducing a word to its root form. For instance, the words 'simpler' and 'simplest' are reduced to 'simple' when stemmed. The aim of stemming is to ensure that words carrying the same meaning (but written in different forms) are transformed to the same format so as to unify frequency counts. We used the Snowball Stemmer because it resulted in a better performance when compared to the Porter Stemmer.

Stop word removal. In NLP tasks, stop words are words that are known to occur more frequently in a language than other words. In many NLP tasks, stop words are usually filtered out since their presence biases the model. In this work, we deduced a corpus specific list of stop words which is based on the frequency of words in the dataset. This implies that words frequently occurring in the corpus were filtered out making our model more robust in handling datasets from different sources.

Corpus specific keywords are the terms with the highest frequency in the dataset. For instance terms that occurred in more than 20% of the dataset were considered stop words because they don't add huge information to the classifier. Some of them ended up being the usual well known stop words like 'the', 'a' and others were just common expressions within the dataset (e.g. 'RT' meaning a retweet from the twitter corpus). The percentage used (20%) was deduced experimentally by testing different ranges and sticking with a value that performed best. This was also useful to reduce the feature space.

3.3 Description of Classifier Features

To generate the feature vectors, different feature extraction techniques were used. Features were determined from emoticons, lexicons, tweet content, POS tags present etc. Details of the features can be found below. It should be noted that unigram features here are single tokens while bigrams are two tokens that are together in a data instance. For example, unigrams of the sentence "I am happy" are ['I', 'am', 'happy'] while it's bigrams are ['I am', 'am happy'].

Lexicon based Features. Five distinct opinion lexicons were used similar to the work of [21]. Two of them were manually generated while the remaining three were created using Distant Supervision. The features extracted from each lexicon for the tweets are; number of positive tokens, score of the maximum scoring token, score of last token and net score of tweet using the sum of the score of its tokens. The lexicons are outlined subsequently:

1 *Bing Liu's Opinion Lexicon:* This is a manually created lexicon with 2,006 positive words and 4,783 negative words. It includes common wrongly spelt terms, slangs and social media lingo making it more valuable than a pure English lexicon. The lexicon was compiled from 2004 to 2012 by [11].

2 *Multi Perspective Question Answering (MPQA) Subjectivity Lexicon:* It contains 8,221 manually labelled unigrams. It indicates the prior polarity of a word alongside its part of speech information [35].

3 *NRC Word-Emotion Association Lexicon:* A unigram lexicon with 14,200 unique words manually labelled as positive or negative.

4 *Sentiment140 Lexicon:* It was automatically generated from Twitter data (1.6million tweets) using Distant Supervision. The lexicon contains 62,468 unigrams and 677,698 bigrams.

5 *NRC Hashtag Sentiment Lexicon:* It was generated using a similar technique to the Sentiment140 lexicon. It contains 54,129 unigrams and 316,531 bigrams.

Emoticon Features. Three distinct features were generated based on emoticons. Two of the features are binary features that indicate the presence or absence of positive or negative emoticons in the tweets. The presence of the desired property sets the feature to 1, while it's absence sets it to 0. The third emoticon-based feature sets a binary feature to 1 or 0, if the tweet ends with a positive or negative emoticon, respectively. The last token of a tweet is significant because it provides valuable insight on the concluding message of a tweet.

POS Tagging. POS tagging involves the assignment of Part of Speech information to a word in text. It is known in NLP circles that part of speech information provides important insight into sentiment information in text. However, POS tagging of tweets using traditional taggers tends to lead to unusual results because of the noise and abundance of out-of-vocabulary (OOV) terms present in tweets. To augment the NLTK Tagger [6], a POS tagger aware of the nature of Twitter lingo is used. The authors of [23] implemented a Twitter aware POS tagger trained with manually labelled POS tagged tweets. After successfully retrieving the POS tags for each tweet, for each tag name in the tag set, the number of times each POS tag occurs is identified and accounted for by an integer value.

Sentence Level Features. Sentence level features accounted for are: upper case word count, elongated word count and presence of punctuations.

- In each tweet, the number of words that appear in uppercase was counted.
- Counting the number of words containing a character sequence greater than two identified elongated word count.
- A binary feature was used to represent if the last token in a tweet is an exclamation or question mark.
- The number of continuous sequence of exclamation or question mark. Negation was handled using the method proposed by [25] which is defined as the region of a tweet that starts with a negation term and ends with any of the punctuations full stop, comma, question mark, colon, semi colon or exclamation mark.

4. Evaluation and Discussion

After implementing the raw ME classifier with default classifier parameters, we obtained an F-Score of 64.62% which served as our baseline paradigm. Following the baseline results, we performed further experiments, using pre-processing techniques and feature extraction methods detailed earlier. We also tuned our classifier parameters and we experimentally identified the optimal combination of features resulting in the best performance. Optimal performance was achieved with the following parameters: *C = 1.47, penalty = l1* (norm used in penalization) and *tol=0.6e-3* (tolerance for stopping).

The following table shows the effect on the classifier's F-score of removing one of the features while keeping the others. From the results, it can be observed that the features based on the Twitter aware POS tagger [23] had the highest positive impact on the F-Score followed by stemming, both increasing the F-Score by a cumulative 3.46%. We experimented with a traditional POS tagger and it skewed the results by reducing the F-Score. This further reinforces the impact of using tools suitable for the short message domain. The use of normalization and removal of stop words during the pre-processing phase boosted the F-Score by a total of 1.62%. The introduction of some of these features resulted in the better performance of our classifier when compared to related work [21], where the features were not used. Stop word removal involved identifying domain-specific stop words based on word frequencies in the data set. Although the lexicon based features improved the

Table 1. Effect of individual features on F-Score for ME.

Experimentation	**F-Score (% difference)**
Optimal features combination	73.59
Part of Speech (POS) Tagging	71.59 (2.00)
Stemming	72.13 (1.46)
Stop word removal	72.27 (1.32)
Negation Handling	72.52 (1.07)
All Lexicons	72.82 (0.77)
Sentence level features: (Capitalization, term elongation, punctuation, emoticons)	73.18 (0.41)
Bigrams	73.27 (0.32)
Normalization	73.28 (0.31)

F-Score by a total of 0.77% it was not as effective as it was in the work of [21] where it resulted in approximately 8% increase. This can be explained by the use of a different ML algorithm in our research and the introduction of novel pre-processing techniques. The test set was the same as the one used in the work of [3] where an F-Score of 68.8% was obtained. Compared to the 73.59% obtained in this work we deduce that the current classifier achieved a percentage increase of 6.96%.

The current work is similar to [21] in the sense that we had intersections in our feature extraction techniques. For instance, we got our lexicons from the same source, used identical datasets and used some similar sentence level features (e.g. number of capitalised words and presence of emoticons). One primary difference between both works is that [21]'s endgame is to sentiment polarity classification while the aim of this research is to make the output of a sentiment analysis system useful to a forensic investigator by making it easy to extract insights from the results (using the Forensic Tool). Furthermore, the machine learning algorithm used for classification in both works was different. In [21] the authors used an SVM while our algorithm was the Logistic Regression based classifier.

It is noteworthy that the test data set does not contain neutral instances. This is because this work aims to help forensic analysts identify fluctuations in emotions the most important being positive to negative or vice versa. However, for the purpose of experimentation, we ran some tests with neutral SMS instances, and the F-Score dropped by 3.6% which is still higher than the result from [3]. This result is also better compared to our previous work featuring a hybrid classifier [4] which reached an accuracy of 62% for three-class classification. Moreover, the F-score of the scheme we present in this paper approximates the cur-

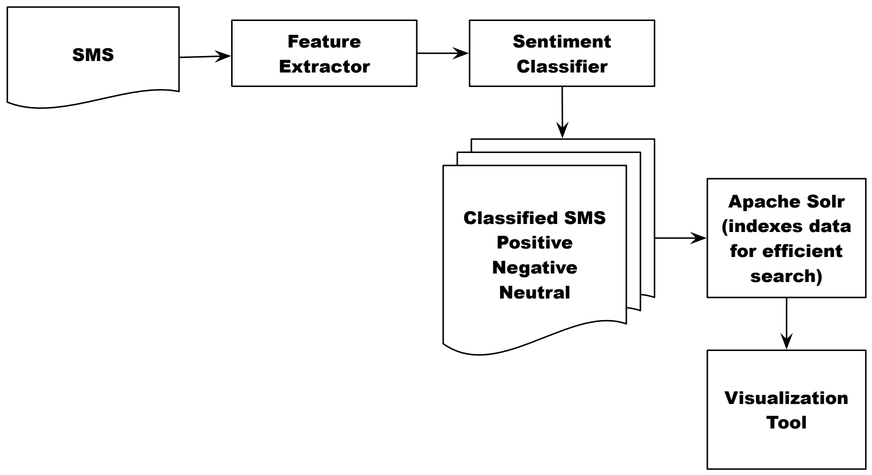

SMS
Feature Extractor
Sentiment Classifier
Classified SMS Positive Negative Neutral
Apache Solr (indexes data for efficient search)
Visualization Tool

Figure 1. Architecture of the Forensic (Sentiment Visualization) Tool.

rent best score (70.28%) [27] which was achieved with an SVM classifier. However, ME models are known to be faster than SVM models, thus we conclude that our classifier is competitive compared to the existing systems.

5. Sentiment Visualization Tool

A web visualization tool was implemented as a proof of concept creating an easy to use interface to extract relevant sentiment information from SMS (Figure 1). The implementation was done using the Python 'Flask' library and the 'Bootstrap' framework for the frontend. The classifier is trained with the set of features resulting in the best F-Score and it is used to predict the sentiment of unknown SMS instances of individuals. Note that despite the classifier was trained with tweets and not SMS messages, the visualization tool uses SMS as a test case. This is because tweets and SMS share a striking similarity in terms of structure. Both formats set restrictions to their length using character limits and they also include words and symbols with common characteristics (e.g. emoticons). More details on the similarity between SMS and tweets can be found in our previous work [3]. Furthermore, the classifier's test results on an unseen SMS dataset presented in Table 1 show that the classifier performs well on SMS datasets.

The messages used to showcase the forensic tool were extracted from the NUS SMS dataset [8]. The version we used contained 45,062 messages sent by over 100 people spanning over 3 years (from 2010 to 2014).

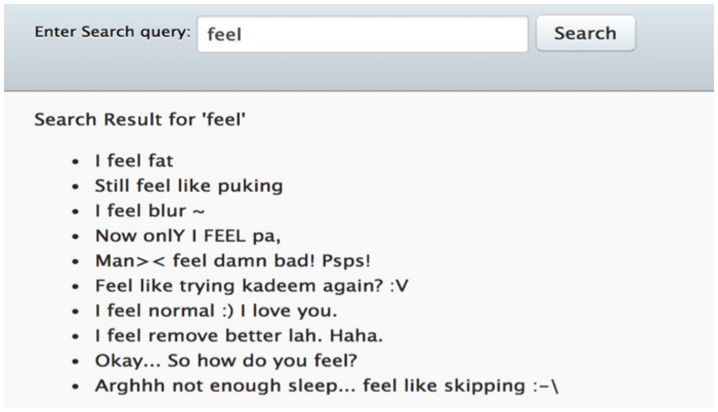
The messages were in XML format and each message tag contained meta-data about the SMS. However, the new version of the dataset contains anonymous information. Each message tag contained the following information: 1) Sender phone number, 2) Recipient phone number, 3) Time message was sent, 4) Age of sender, 5) Country and city where message was sent.

After parsing the XML message data, the sender, recipient and time fields were retrieved for each SMS message. The age, country and city fields were not used in this research work. Each SMS was then pre-processed using the same pre-processing techniques utilized when training the classifier. Moreover, features were extracted and fed into the classifier as test input data for sentiment polarity classification. The classifier outputs the polarity of each SMS message and the classified messages are pushed into Apache Solr [29] for storage and indexing.

Apache Solr is a fast, open source enterprise search software built on Apache Lucene, used in our previous work [3]. Solr allows facetted search, which entails dynamic clustering of search results to help users easily drill down to answers they want. An example of a facetted search in the context of this project is to find messages that have a negative polarity, and are sent by a particular user S after a given time T. The ability to have such strong grip on the data retrieval process is the rationale behind pushing the data into Solr. Additional interesting features can be built into the forensic tool in the future because of the features provided by Solr.

Once the visualization software interface is launched, it accesses the relevant Solr core. The Solr core is the running instance of Lucene (with the Solr configuration) and gets information about the people that communicated with the person under investigation. These entities are pre-loaded into a dropdown list. The person of interest can thus be selected and information about the polarity of messages sent by the individual can be visualized. The preloaded data creates an avenue to showcase the features of our forensic tool.

In a real life use case scenario we would expect to follow a certain procedure during a forensic analysis: 1) Obtain a physical image from the device, 2) fetch SMS from the SQLite database (mmssms.db if the mobile device runs under the Android OS), c) classify them with the trained classifier and d) push the result into Solr, where the visualization tool can access it. This research work does not cover techniques for extracting messages from a mobile device. Information on extracting physical digital images and SMS messages from Android devices is detailed in [2] and [3]. The visualization tool consists of the following



Search Result for 'feel'

- I feel fat
- Still feel like puking
- I feel blur ~
- Now onlY I FEEL pa,
- Man>< feel damn bad! Psps!
- Feel like trying kadeem again? :V
- I feel normal :) I love you.
- I feel remove better lah. Haha.
- Okay... So how do you feel?
- Arghhh not enough sleep... feel like skipping :-\

Figure 2. Screenshot of Forensic Tool Search component.

components: a) Pie chart, b) Tag Cloud, c) Sentiment Timeline View and d) Search Interface.

Search Tool. A search tool was implemented allowing a user to search for an occurrence of any desired term within the messages. As a use case scenario, we assume that there is an interest to find out the messages, where the user mentions the word ‘feel’. The search box seen in Figure 2 can be used to enter a search query. Also Figure 2 shows the output with relevant results. While the search tool comes in handy when an analyst knows what to look for, it is not very helpful in scenarios where there is no prior knowledge of keywords that reveal interesting patterns. To solve this problem, we created the sentiment timeline view (discussed later in this section) that helps an analyst discover patterns and a tag cloud that gives the analyst information on the most common keywords within the messages.

Polarity Distribution View. This visualization component is a pie chart and it is used to show the percentage polarity distribution of the sent or received messages. Figure 3 illustrates a screenshot that shows the polarity distribution of sent messages for a given user as seen on the dashboard of the sentiment visualization tool.

Tag Cloud. A tag cloud is used to render the most common words in messages with negative or positive polarity. This gives an observer a feel of terms that are often associated with a specific emotion by the individual. The tag cloud we implemented is interactive as it responds to mouse click events. When a word is clicked in the tag cloud, messages

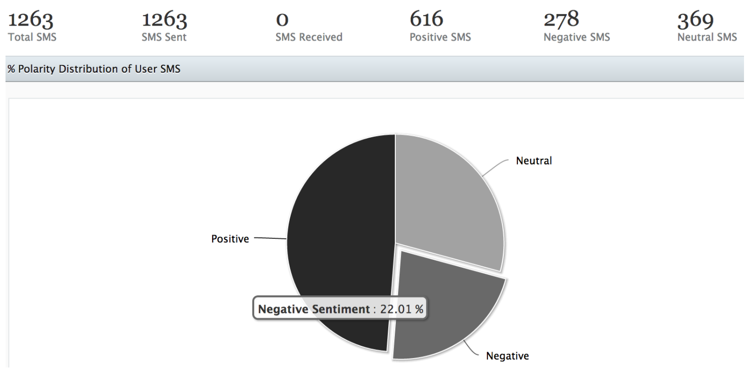


Figure 3. Screenshot of Polarity Distribution of suspect SMS messages represented in a pie chart.

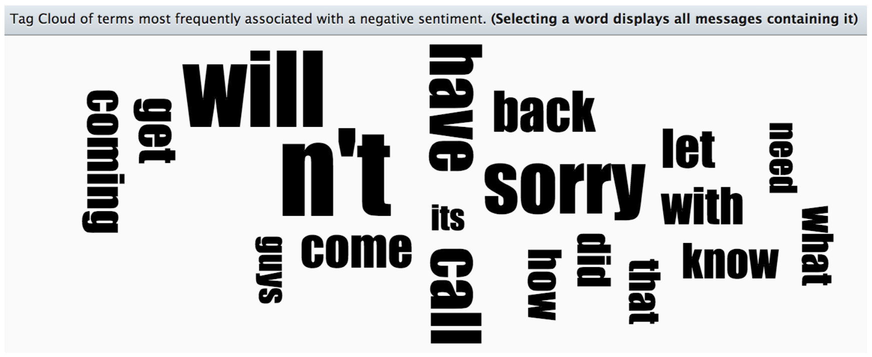

Tag Cloud of terms most frequently associated with a negative sentiment. (Selecting a word displays all messages containing it)
coming
get
will
n't
guys
come
have
its
call
back
sorry
how
did
that
let
with
know
need
what

Figure 4. Screenshot of Tag Cloud of suspect SMS in Visualization Tool.

containing the word are returned. Figure 4 shows a screenshot of the tag cloud generated for a sample user.

Sentiment Timeline View. A Sentiment Timeline View (first presented in [3]) was implemented to ease analysis of the mood swing of an individual over time. Figure 5 shows a screenshot of the timeline view with the horizontal axis representing the day and the vertical axis representing number of messages sent. The Sentiment Timeline View lies at the heart of the visualization tool as it provides insight on the emotional swing of an individual in an automated fashion.

When the mouse cursor hovers on a node, a tooltip is used to display the number of SMS messages the node represents. The node can then be clicked to view the content of the messages sent. As seen in the screenshot, on Friday 12th March, it can be observed that the user experienced a sudden spike in emotional swing. This is because the user

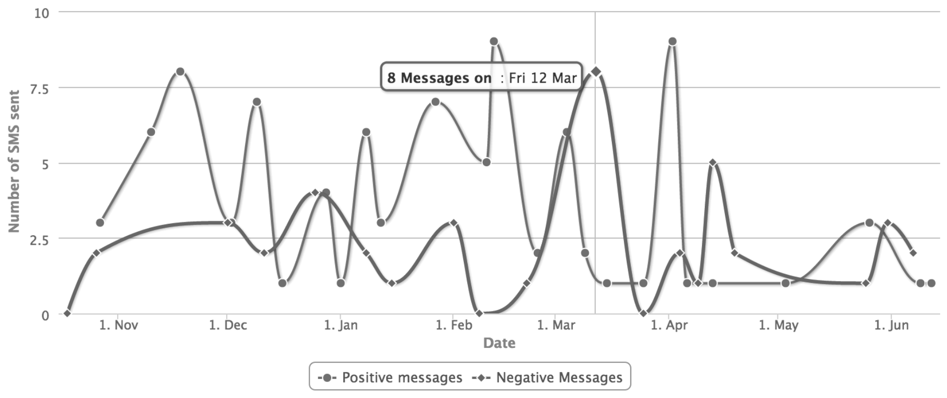


Figure 5. Screenshot of Sentiment Timeline component of Forensic tool.

sent 8 negative messages that day but did not send any negative message the previous day. The forensic tool extracts patterns of this nature and it can help the forensic analyst identify emotional fingerprints that will have been otherwise hidden. This feature is more important than the search feature because it reveals insights that the forensic expert can not anticipate through keyword based search. It therefore provides valuable information on emotionally volatile periods of the subject under investigation.

6. Conclusion and Future Work

This research work has succeeded in bringing the problems plaguing sentiment analysis in the short message domain to the fore and presents useful tools to solve each of those problems. A sentiment aware tokenizer was used, a POS-Tagger suitable for the short message domain was utilized, normalization was implemented and negation was handled. Amidst all the features used, the POS tagging feature proved to be most effective followed by stemming. The use of normalization, domain specific stop words (based on term frequency) and bigram features absent in related work further improved the results. We also showed that the classifier performed well on an SMS test set, validating the similarity between SMS and tweets and affirming that the model does not over-fit. Experimentation was done with several sentence level features and the effect of normalization in Sentiment Polarity Classification was demonstrated.

An intuitive visualization tool was implemented to aid extraction of intelligence information regarding an individual. It helps visualize the mood swing of an individual overtime and such information can prove useful for a forensic analyst. The ‘Timeline View’ of the sentiment visualization tool presents an improved approach to identify periods of emotional instability within the messages sent by a subject. This presents a unique automated view into understanding the sentiment expressed by an individual over time. The tool allows intuitive visualization of the mood swing of an individual providing valuable patterns of low times and high times. This tool does not in any way aim to replace a forensic analyst; it simply serves as an additional source of information for further investigation.

This research shows potential for developing a forensic tool based on SMS information retrieved from a user’s mobile device combining all aspects of our previous works [3, 4]. The visualization tool can be further improved to provide an overview of the subjects (or topics) discussed in the messages. A keyword based preliminary version of this is already achieved by the tag cloud. This will entail providing an overview of the summary of a group of messages. The current tool shows how the emotional state of an individual evolves over time based on sent messages.

To further improve the classification F-Score and efficiency, well-established feature reduction techniques like Principal Component Analysis (PCA) can be used to reduce the feature space. This will improve the runtime of the classifiers and it may improve classification results. The classifier powering the forensic tool uses the default decision threshold for classification. ROC analysis can be done to identify the optimal threshold for each of the classifiers so as to improve the classification accuracy.

References

[1] O. Aboluwarin, Sentiment Analysis of Short Messages (Tweets, SMS), (`github.com/Pelumi/ShortMsgAnalysis`), 2016.

[2] P. Andriotis, G. Oikonomou and T. Tryfonas, Forensic analysis of wireless networking evidence of Android smartphones, *Proceedings of the Fourth IEEE International Workshop on Information Forensics and Security (WIFS’12)*, pp. 109–114, 2012.

[3] P. Andriotis, A. Takasu and T. Tryfonas, Smartphone message sentiment analysis, in *Advances in Digital Forensics X*, G. Peterson and S. Shenoi (eds.), pp. 253–265, Springer Berlin Heidelberg, 2014.

[4] P. Andriotis and G. Oikonomou, Messaging activity reconstruction with sentiment polarity identification, in *Human Aspects of Information Security, Privacy, and Trust (HAS 2015)*, T. Tryfonas and I.G. Askoxylakis (eds.), pp. 475–486, Springer International Publishing, 2015.

[5] N.L. Beebe and J.G. Clark, Digital forensic text string searching: Improving information retrieval effectiveness by thematically clustering search results, *Digital investigation*, vol. 4, pp. 49–54, 2007.

[6] S. Bird, Nltk: the natural language toolkit, *Proceedings of the COLING/ACL on Interactive Presentation Sessions*, pp. 69–72, 2006.

[7] E. Cambria, B. Schuller, Y. Xia and C. Havasi, New avenues in opinion mining and sentiment analysis, *IEEE Intelligent Systems*, vol. 28 (2), pp. 15–21, 2013.

[8] T. Chen and M.Y. Kan, Creating a live, public short message service corpus: The nus sms corpus, *Language Resources and Evaluation*, vol. 47 (2), pp. 299–335, 2013.

[9] N. Cheng, R. Chandramouli and K. Subbalakshmi, Author gender identification from text, *Digital Investigation*, vol. 8 (1), pp. 78–88, 2011.

[10] O. De Vel, A. Anderson, M. Corney and G. Mohay, Mining e-mail content for author identification forensics, *ACM Sigmod Record*, vol. 30 (4), pp. 55–64, 2001.

[11] X. Ding, B. Liu and P.S. Yu, A holistic lexicon-based approach to opinion mining, *Proceedings of the First ACM International Conference on Web Search and Data Mining*, pp. 231–240, 2008.

[12] M. Gamon, Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis, *Proceedings of the Twentieth International Conference on Computational Linguistics*, pp. 841, 2004.

[13] A. Go, R. Bhayani and L. Huang, Twitter sentiment classification using distant supervision, *CS224N Project Report*, Stanford, pp. 1–12, 2009.

[14] B. Han, P. Cook and T. Baldwin, Lexical normalization for social media text, *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 4 (1), pp. 1–27, 2013.

[15] F. Iqbal, H. Binsalleeh, B. Fung and M. Debbabi, Mining writeprints from anonymous e-mails for forensic investigation, *Digital investigation*, vol. 7 (1), pp. 56–64, 2010.

[16] X. Jin, Y. Li, T. Mah and J. Tong, Sensitive webpage classification for content advertising, *Proceedings of the First ACM International Workshop on data mining and audience intelligence for advertising*, pp. 28–33, 2007.

[17] P. Juola, Authorship attribution, *Foundations and Trends in information Retrieva*, vol. 1 (3), pp. 233–334, 2006.

[18] C. Kobus, F. Yvon and G. Damnati, Normalizing sms: are two metaphors better than one?, *Proceedings of the Twenty-Second International Conference on Computational Linguistics*, pp. 441–448, 2008.

[19] W. Ling, C. Dyer, A.W. Black and I. Trancoso, Paraphrasing 4 microblog normalization, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 73–84, 2013.

[20] E. Martínez-Cámara, M.T Martín-Valdivia, L.A. Ureña-López and A.R. Montejo-Ráez, Sentiment analysis in twitter, *Natural Language Engineering*, vol. 20 (1), pp. 1–28, 2014.

[21] S.M. Mohammad, S. Kiritchenko and X. Zhu, Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets, *Proceedings of the Second Joint Conference on Lexical and Computational Semantics*, pp. 321–327, 2013.

[22] P. Nakov, S. Rosenthal, Z. Kozareva, V. Stoyanov, A. Ritter and T. Wilson, SemEval-2013 Task 2: Sentiment Analysis in Twitter, (`www.cs.york.ac.uk/semeval-2013/accepted/101_Paper.pdf`), 2013.

[23] O. Owoputi, B. OConnor, C. Dyer, K. Gimpel, N. Schneider and N.A. Smith, Improved part-of-speech tagging for online conversational text with word clusters, *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 380–391, 2013.

[24] B. Pang and L. Lee, Opinion mining and sentiment analysis, *Foundations and trends in information retrieval*, vol. 2 (1-2), pp. 1–135, 2008.

[25] B. Pang, L. Lee and S. Vaithyanathan, Thumbs up?: sentiment classification using machine learning techniques, *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pp. 79–86, 2002.

[26] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, Scikit-learn: Machine learning in python, *The Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[27] S. Rosenthal, A. Ritter, P. Nakov and V. Stoyanov, SemEval-2014 Task 9: Sentiment Analysis in Twitter, (`www.aclweb.org/anthology/S/S14/S14-2.pdf#page=93`), 2014.

[28] Sentiment Analysis in Twitter, (`www.cs.york.ac.uk/semeval-2013/task2`), 2013.

[29] Solr project homepage, (`lucene.apache.org/solr/`), 2016.

[30] A. Stolerman, R. Overdorf, S. Afroz, R. Greenstadt, Breaking the closed-world assumption in stylometric authorship attribution, in *Advances in Digital Forensics X*, G. Peterson and S. Shenoi, (eds.), pp. 185–205, Springer Berlin Heidelberg, 2014.

[31] L.V. Subramaniam, S. Roy, T.A. Faruquie and S. Negi, A survey of types of text noise and techniques to handle noisy text, *Proceedings of the Third ACM Workshop on Analytics for Noisy Unstructured Text Data*, pp. 115–122, 2009.

[32] J. Suttles and N. Ide, Distant supervision for emotion classification with discrete binary values, in *Computational Linguistics and Intelligent Text Processing*, A. Gelbukh (ed.), pp. 121–136, Springer Berlin Heidelberg, 2013.

[33] P.D. Turney, Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews, *Proceedings of the Fortieth annual meeting on association for computational linguistics*, pp. 417–424, 2002.

[34] H. Wang, D. Can, A. Kazemzadeh, F. Bar and S. Narayanan, A system for real-time twitter sentiment analysis of 2012 us presidential election cycle, *Proceedings of the ACL 2012 System Demonstrations*, pp. 115–120, 2012.

[35] T, Wilson, J. Wiebe and P. Hoffmann, Recognizing contextual polarity in phrase-level sentiment analysis, *Proceedings of the conference on human language technology and empirical methods in natural language processing (HLT-EMNLP-2005)*, pp. 347–354, 2005.