# Optimizing Short Message Text Sentiment Analysis for Mobile Device Forensics

Oluwapelumi Aboluwarin, Panagiotis Andriotis, Atsuhiro Takasu, Theo Tryfonas



Chapter 4

# OPTIMIZING SHORT MESSAGE TEXT SENTIMENT ANALYSIS FOR MOBILE DEVICE FORENSICS

Oluwapelumi Aboluwarin, Panagiotis Andriotis, Atsuhiro Takasu and Theo Tryfonas

**Abstract** Mobile devices are now the dominant medium for communications. Humans express various emotions when communicating with others and these communications can be analyzed to deduce their emotional inclinations. Natural language processing techniques have been used to analyze sentiment in text. However, most research involving sentiment analysis in the short message domain (SMS and Twitter) do not account for the presence of non-dictionary words. This chapter investigates the problem of sentiment analysis in short messages and the analysis of emotional swings of an individual over time. This provides an additional layer of information for forensic analysts when investigating suspects. The maximum entropy algorithm is used to classify short messages as positive, negative or neutral. Non-dictionary words are normalized and the impact of normalization and other features on classification is evaluated; in fact, this approach enhances the classification F-score compared with previous work. A forensic tool with an intuitive user interface has been developed to support the extraction and visualization of sentiment information pertaining to persons of interest. In particular, the tool presents an improved approach for identifying mood swings based on short messages sent by subjects. The timeline view provided by the tool helps pinpoint periods of emotional instability that may require further investigation. Additionally, the Apache Solr system used for indexing ensures that a forensic analyst can retrieve the desired information rapidly and efficiently using faceted search queries.

**Keywords:** Sentiment analysis, text mining, SMS, Twitter, normalization

## 1. Introduction

The ubiquity of mobile devices has redefined the communications landscape around the world. This has led to the creation of valuable individual data through conversational services such as SMS and micro blogging platforms such as Twitter. Mining the content of these interactions can provide valuable insights into the communicating entities. Information such as the time that an interaction occurred and the content of the communication can be useful to forensic analysts because it can reveal patterns that are hidden in the text.

Additional information about the disposition of conversations can be extracted using machine learning techniques. Sentiment analysis is the discipline concerned with retrieving opinion or emotion expressed in text. In the research literature, applications of sentiment analysis have been proposed in a variety of fields, especially related to social media and micro blogging services. As discussed in [1, 3], sentiment information can also be useful in forensic investigations of smartphones.

This chapter investigates the use of sentiment analysis to model the emotional swings of an individual as opposed to the emotional swings of a group of people towards a brand, which is more common in the research literature. Machine learning algorithms are employed for sentiment polarity classification. Normalization is used to account for lexically-incorrect terms that are prevalent in conversational texts; these invalid terms are known to negatively impact the efficiency of natural language processing [21]. A forensic tool with an intuitive user interface has been developed to support the extraction of sentiment information. The emotional timeline generated by the tool provides an additional layer of information about a person under investigation because it helps a forensic analyst identify periods of time during which the individual exhibited a volatile emotional state.

This research has several key contributions. First, the normalization of non-dictionary words alongside other sentence-level features is shown to improve sentiment polarity classification. Furthermore, the incorporation of a part-of-speech tagger (POS-Tagger) that is aware of the peculiarities of short messages enhances classifier performance. Another contribution is the analysis of how individual features affect the performance of the most efficient classifier of emotions in short text messages (SMS). Finally, the implemented forensic tool provides details about sentiment polarity expressed in an individual's SMS messages in a concise and intuitive manner to facilitate the rapid extraction of information by forensic analysts (see `github.com/Pelumi/ShortMsgAnalysis`).

## 2. Related Work

The need to know the opinions of others on subjects of interest is valuable when attempting to make decisions in an unfamiliar terrain [6, 19, 22, 27]. The ubiquity of online reviews and recommendations makes the web a go-to place for individuals who seek such information. People rely on opinions posted on the web to guide decisions about products, movies, employers, schools, etc. Increased interest in this sort of information has been the major driver of research in sentiment analysis. Sentiment analysis started receiving increased attention in the research landscape in 2002 (see, e.g., [23, 27]) and has since been studied extensively, leading to its use in applications such as content advertising [15], election monitoring [29] and customer feedback data classification [11].

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 [22]. Since some text instances are neither positive nor negative, sentiment analysis also involves identifying texts that do not convey any form of emotion (these are referred to as "neutral"). Hence, sentiment analysis problems are handled as classification or regression tasks. 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, sentiment analysis problems can be treated as multi-class classification tasks when the instances to be classified fall under categories such as positive, negative and neutral.

Sentiment analysis techniques include: (i) lexicon-based methods [3]; (ii) machine learning methods [1]; and (iii) hybrid approaches that combine lexicon-based and machine learning methods [1, 10]. When treating sentiment analysis as a classification task, machine learning algorithms that are known to perform well in text classification are often used. Some of the supervised learning algorithms commonly used in the literature are support vector machines (SVMs), multinomial naive Bayes (MNB) and maximum entropy (logistic regression) [12, 23, 26].

In digital forensics, text mining methods have been used for tasks such as authorship attribution in email [16] and text string search [4]. Support vector machine algorithms have also been used for email authorship attribution [9] and to identify the genders of the authors of SMS texts [8]. Authorship attribution experiments have also been conducted using machine learning techniques [14, 25]. The work of Mohammad et al. [20] is closely related to this research because it focuses on extracting sentiment polarity information from Twitter feeds (tweets). The work details the techniques used by the research team that obtained the high-

est accuracy (F-score) at the SemEval-2013 Task 2: Sentiment Analysis in Twitter Competition for Sentiment Polarity Classification.

Andriotis et al. [3] have applied sentiment analysis to augment digital forensic investigations by retrieving opinion information from SMS texts found in mobile devices. A lexicon-based technique was used for sentiment polarity classification and a proof-of-concept system was developed to visualize mood patterns extracted from SMS message databases. The maximum classification accuracy achieved was 68.8% (for positive SMS messages). The classification F-scores were improved in [1] and an F-score of 74.4% was obtained for binary classification (SMS: positive superclass and negative class). However, this work included neutral and positive messages in a superclass, which resulted in large false-positive rates. These error rates were decreased dramatically with a hybrid classifier [1], but the total estimated F-score also decreased (62%) when a three-class categorization (positive, neutral, negative) was performed. The best sentiment classification performance for SMS so far was achieved in SemEval-2014 Task 9: Sentiment Analysis in Twitter Contest. The winning team obtained an F-score of 70.28% for classifying SMS texts in three classes, but no published information is available about the false-positive rate.

Since machine learning techniques are known to outperform lexicon-based methods [12], this research focused on the use of machine learning methods for sentiment classification in an attempt to enhance the accuracy of the forensic tool. The research also has drawn cues from sentence-level features presented in [20]. However, the classification results have been improved by integrating the normalization of non-dictionary words. The work of Venkata Subramaniam et al. [28], which analyzes commonly-used techniques for handling noisy text, was also leveraged in this research. Finally, the statistical machine translation (SMT) technique for normalization presented in [13] served as the basis of the normalization task.

## 3. Datasets and Classification

The classifier was trained using the multinomial logistic regression algorithm, also known as the maximum entropy (ME) algorithm. The maximum entropy algorithm makes it possible to apply logistic regression to multi-class classification problems like the three-class short message classification task considered in this work. Maximum entropy is usually preferred over the multinomial naive Bayes (MNB) algorithm because it does not assume the statistical independence of features. Therefore, it implicitly takes natural language processing properties like nega-

tion into consideration when creating models. While the training time for the maximum entropy algorithm is somewhat higher than that for the multinomial naive Bayes algorithm, the training time is much lower than those for other algorithms such as support vector machines [20]. The maximum entropy algorithm used in this work was implemented in Python using the `scikit-learn` library [24]. Parameter tuning was carried out by a `scikit-learn` process called Grid Search, which involves the specification of a range of parameters and allowing the system to run through the permutations to identify the optimal combination.

## 3.1 Datasets

The dataset used during the SemEval-2013 competition was utilized for training the models (`www.cs.york.ac.uk/semeval-2013/task2`). The training dataset contained 8,120 tweets (positive: 37.2%, negative: 14.7% and neutral: 48.1%). The testing dataset from [3] was also employed, making it possible to compare the results directly.

## 3.2 Pre-Processing

Pre-processing involves the cleaning of a raw dataset before applying a machine learning algorithm. It is a standard procedure in most machine learning tasks and the techniques used vary across domains. Pre-processing ensures that noisy data is in a proper shape for the application of machine learning algorithms. In text mining, pre-processing often involves normalization, spelling correction, managing text encoding, etc. Some of the techniques used in this research are described below.

- **Normalization:** In this context, normalization involves resolving lexically-incorrect monosyllabic terms to their correct form. The terms may be in the form of spelling mistakes or *ad hoc* social media short forms as defined in [18]. Normalization is known to improve the quality of some natural language processing tasks such as language translation [17, 18]. The normalization used in this research involved statistical machine translation; some of the techniques used are described in [18]. The outcome of the normalization task is a dictionary mapping of lexically-incorrect terms to their lexically-correct variants. An example is the mapping of each word in "raw text" to the corresponding word in "normalized text" in the following representation:
  - **Raw Text:** Hi ranger hw r u
  - **Normalized Text:** Hi ranger how are you

Statistical machine translation requires a parallel corpus – a list of messages containing lexically-incorrect terms mapped to their lexically-correct forms. In the dataset used in this research, the total number of "incorrect terms" mapped to "corrected terms" using statistical machine translation was 156. Thus, the generated normalization dictionary was quite small due to the limited size of the corpus. To address this disadvantage, the normalization dictionary in [13] containing more than 41,181 normalization candidates in the short message domain was also employed.

To apply the normalization dictionary to the corpus, each tweet was tokenized and lexically-correct tokens were filtered, leaving only lexically-invalid tokens. The lexically-correct terms were then identified based on their presence in an English dictionary using the Python Enchant spell-checking library; Enchant helps identify words that are not in the dictionary of a defined language (interested readers are referred to `bit.ly/pyench` for additional details). The remaining lexically-correct terms were identified by checking for their presence in online slang dictionaries (e.g., Urban Dictionary). The normalization of data instances before sentiment polarity classification 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 in tweets were replaced with empty strings. In addition, occurrences of the term "RT," which means retweet on Twitter, were removed. These terms were removed to prevent the over-fitting of the model on the Twitter dataset (mentions, retweets and URLs are not as common in SMS texts as they are in tweets). Positive emoticons were replaced with words known to have positive connotations while negative emoticons were replaced with negative polarity words. This ensured that the information added by emoticons to the model was not lost during the tokenization process, since emoticons are prone to ambiguous tokenization.

  Data cleaning also involved the unification of elongated expressions. In this research, elongated expressions are terms with a sequence of three or more characters (e.g., "whyyyy"). These expressions are commonly used to convey emphasis in social media and the number of elongated characters varies across users. All elongated characters were trimmed to a maximum of two characters (e.g., "killll" was trimmed to "'kill"). This makes it easier to identify words that convey the same emotion.

- **Stemming:** This process reduces a word to its root form. For example, the words "simpler" and "simplest" are reduced to "simple" when stemmed. The goal of stemming is to ensure that words that carry the same meaning (but written in different forms) are transformed to the same format in order to unify their frequency counts. The Snowball Stemmer was used in this research because it exhibits better performance than the Porter Stemmer.

- **Stop Word Removal:** Stop words are words that are known to occur more frequently in a language than other words. In many natural language processing tasks, stop words are usually filtered because their presence biases the model. In this work, corpus-specific stop words were deduced based on the frequencies of the words in the dataset. Thus, frequently-occurring words in the corpus were filtered to make the model more robust in handling datasets from different sources.

  Corpus-specific keywords are the terms with the highest frequencies in a dataset. For example, terms that occurred in more than 20% of the dataset were considered stop words because they do not add much information to the classifier. Some of them were common stop words (e.g., "the" and "a") and others were just common expressions in the dataset (e.g., "RT" corresponding to retweet in the Twitter corpus). The percentage used (20%) was deduced experimentally by testing different ranges and sticking with the value that performed best. This also helped reduce the feature space.

## 3.3 Classifier Features

Various feature extraction techniques were used to generate the feature vectors. The features were determined from emoticons, lexicons, tweet content, part-of-speech tags present, etc. Details of the features are provided below. Note that unigram features correspond to single tokens while bigrams are two tokens that appear together in a data instance. For example, unigrams of the sentence "I am happy" are ["I," "am," "happy"] while the bigrams are ["I am," "am happy"].

- **Lexicon-Based Features:** Five distinct opinion lexicons were used as in [20]. Two of them were manually generated while the remaining three were created using the distant supervision learning scheme. The features extracted from each lexicon for the tweets were: number of positive tokens, score of the maximum scoring token, score of the last token and net score of a tweet using the sum of the scores of its tokens. The lexicons used were:

  - **Bing Liu's Opinion Lexicon:** This is a manually-created lexicon with 2,006 positive words and 4,783 negative words. It includes common incorrectly-spelled terms, slang and social media lingo, making it more valuable than a pure English lexicon. The lexicon was compiled from 2004 to 2012 [10].
  - **Multi-Perspective Question Answering Lexicon:** This lexicon contains 8,221 manually-labeled unigrams (available at `mpqa.cs.pitt.edu/lexicons/subj_lexicon`). It indicates the prior polarity of a word alongside its part-of-speech information.
  - **NRC Word-Emotion Association Lexicon:** This unigram lexicon has 14,200 unique words manually-labeled as positive or negative.
  - **Sentiment140 Lexicon:** This lexicon was automatically generated from Twitter data (1.6 million tweets) using distant supervision. The lexicon contains 62,468 unigrams and 677,698 bigrams.
  - **NRC Hashtag Sentiment Lexicon:** This lexicon was generated using a similar technique to that used for the Sentiment140 lexicon. It contains 54,129 unigrams and 316,531 bigrams.

- **Emoticon Features:** Three features were generated based on emoticons. Two were binary features that indicate the presence or absence of positive or negative emoticons in tweets. The presence of the desired property sets the feature to one, while the absence sets it to zero. The third emoticon-based feature sets a binary feature to one or zero, if the tweet ends with a positive or negative emoticon, respectively. The last token of a tweet is significant because it provides valuable insights into the concluding message of the tweet.

- **Part-of-Speech Tagging:** This involves the assignment of part-of-speech information to a word in text. In natural language processing circles, it is well known that part-of-speech information provides important insights into sentiment information in text. However, part-of-speech tagging of tweets using traditional taggers tends to yield unusual results due to noise and the abundance of out-of-vocabulary (OOV) terms present in tweets. The NLTK Tagger [5] was augmented with a part-of-speech tagger that was aware of the nature of Twitter lingo. Owoputi et al. [21] have implemented a Twitter-aware part-of-speech tagger trained with

manually-labeled part-of-speech-tagged tweets. After successfully retrieving the part-of-speech tags for each tweet, for each tag name in the tag set, the number of times each tag occurs was identified and accounted for by an integer value.

- **Sentence-Level Features:** The sentence-level features considered in this research were the upper case word count, elongated word count and presence of punctuation.
  - In each tweet, the number of words that appeared in uppercase was counted.
  - The number of words containing a character sequence greater than two (i.e., elongated words) was counted.
  - A binary feature was used to denote if the last token in a tweet was an exclamation point or question mark.
  - The number of continuous sequences of exclamation points or question marks was counted. Negation was handled using the method described in [23]; this is defined as the region of a tweet that starts with a negation term and ends with any of the punctuation marks: period, comma, question mark, colon, semi colon or exclamation point.

## 4. Evaluation and Discussion

The raw maximum entropy classifier with default classifier parameters yielded an F-score of 64.62%, which served as the baseline for the experiments. The experiments were performed using the pre-processing techniques and feature extraction methods discussed above. The classifier parameters were also tuned and the optimal combination of features resulting in the best performance were identified via experimentation. Optimal performance was achieved with the parameters: C = 1.47; penalty = L1 (norm used in penalization) and tolerance = 0.6E-3 (tolerance for termination).

Table 1 shows the impact on the classifier F-score when one of the features is removed while retaining the others. The results indicate that Twitter-aware part-of-speech tagging [21] has the highest positive impact on the F-score followed by stemming, both of them increasing the F-score by a cumulative 3.46%. Experiments were also conducted using a traditional part-of-speech tagger, but it skewed the results by reducing the F-score. This further reinforces the need to use tools that are well suited to the short message domain. The use of normalization and the 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

*Table 1.* Effect of individual features on the F-score.

| **Experiments** | **F-Score % (Difference)** |
|---|---|
| Optimal Features Combination | 73.59 (—) |
| Part-of-Speech 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) |

resulted in better classifier performance compared with related work [20], which did not use the features. Stop word removal involved identifying the domain-specific stop words based on word frequencies in the dataset.

Although the lexicon-based features improved the F-score by a total of 0.77%, they were not as effective as in [20], where they increased the F-score by approximately 8%. This can be explained by the use of a different machine learning algorithm in this research and the introduction of novel pre-processing techniques. The test set used in this research was the same as that used in [3], where an F-score of 68.8% was obtained. Based on the F-score of 73.59% obtained in this work, it can be deduced that the current classifier achieved a percentage increase of 6.96%.

The current work is similar to that of Mohammad et al. [20] due to an intersection in the feature extraction techniques used. In particular, the lexicons came from the same source, identical datasets were used and some similar sentence-level features (e.g., number of capitalized words and presence of emoticons) were employed. However, the primary difference between the two works is that Mohammad et al. [20] focused on sentiment polarity classification while the goal of this research was to make the output of a sentiment analysis system useful to forensic investigators by making it easy to extract insights from the results obtained using the forensic tool. Additionally, the machine learning algorithms used for classification differed. Mohammad et al. [20] used a support vector machine whereas the present work employed a logistic regression based classifier.

It is important to note that the test dataset did not contain neutral instances. This is because the focus was on enabling forensic analysts to identify fluctuations in emotions, the most important being positive

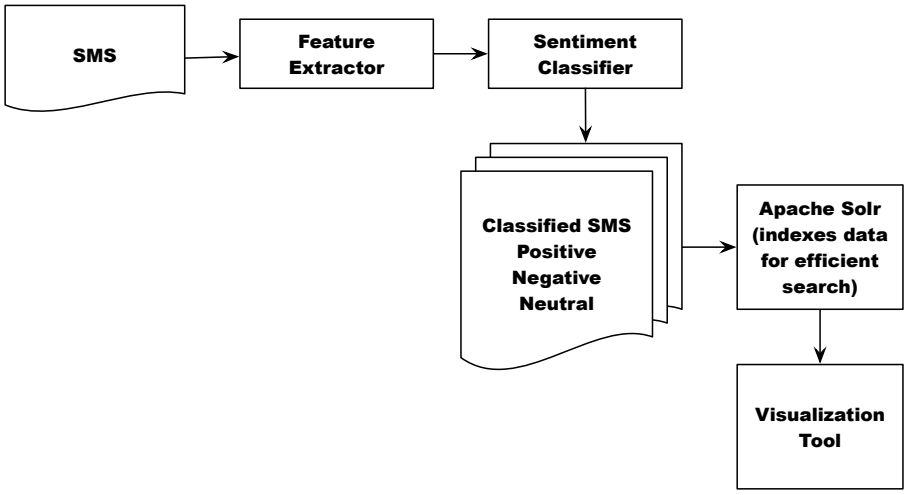


*Figure 1.* Sentiment visualization tool.

to negative sentiments or vice versa. However, when experiments were conducted with neutral SMS instances, the resulting F-score dropped by 3.6%, but this score is still higher than the score reported in [3]. This result is also better that that obtained in previous work featuring a hybrid classifier [1], which yielded an F-score of 62% for three-class classification. Moreover, the F-score (73.59% – 3.6% = 69.99%) obtained in this work approximates the current best score (70.28%) achieved with a support vector machine classifier in the SemEval-2014 Task 9: Sentiment Analysis in Twitter Contest. However, maximum entropy models are known to be faster than support vector machine models. Thus, the classifier presented in this work is competitive compared with existing systems.

## 5. Sentiment Visualization Tool

A web visualization tool was implemented with an easy-to-use interface for extracting relevant sentiment information from SMS texts (Figure 1). The implementation leveraged the Python Flask library and the Bootstrap framework for the front-end. The classifier, which was trained using the feature set that yielded the best F-score, was used to predict the sentiments of SMS texts created by unknown individuals. Note that, although the classifier was trained with tweets, not SMS messages, the visualization tool used SMS messages as a test case. This is because tweets and SMS messages are strikingly similar in terms of structure. Both formats set restrictions on length using character limits

and they also include words and symbols with common characteristics (e.g., emoticons) – interested readers are referred to [3] for details about the similarities between tweets and SMS messages. Furthermore, the test results obtained for the unseen SMS dataset presented in Table 1 demonstrate that the classifier performs well on SMS datasets.

The messages used to showcase the forensic tool were extracted from the NUS SMS dataset [7]. The version used contained 45,062 messages sent by more than 100 people from 2010 to 2014. The messages were in the XML format and each message tag contained metadata about the SMS messages (the new version of the dataset contains anonymous information). Each message tag contained: (i) sender phone number; (ii) recipient phone number; (iii) time message was sent; (iv) sender age; and (v) city and country where the message was sent.

After parsing the XML message data, the sender, recipient and time fields were retrieved for each SMS message. The sender age, city and country fields were not used in this research. Each SMS message was then pre-processed by applying the same techniques that were used when training the classifier. Features were extracted and fed to the classifier as test input data for sentiment polarity classification.

The classifier outputted the polarity of each SMS message and the classified messages were moved to the Apache Solr system for storage and indexing. Apache Solr is a fast, open-source, enterprise search system built on the Apache Lucene system used in previous research [3]. Solr allows faceted search, which involves dynamic clustering of search results to enable users to drill down to the answers they desire. An example of a faceted search in the context of this research is to find messages that have negative polarity and are sent by a particular user $S$ after a given time $T$. The ability to have such a strong grip on the data retrieval process was the rationale for pushing data into Solr. Additional functionality can be built into the forensic tool in the future because of the features provided by Solr.

After the visualization software interface is launched, it accesses the relevant Solr core and provides information about the individuals who communicated with the person under investigation. The names of these individuals are pre-loaded into a dropdown list. An individual of interest can then be selected and information about the polarity of messages sent by the selected individual can be visualized. The pre-loaded data creates an avenue for showcasing the features of the forensic tool.

In a real-world use case, the following steps would be performed during a forensic investigation: (i) obtain a physical image from a mobile device; (ii) fetch the SMS messages from the SQLite database (`mmssms.db` for an Android device); (iii) classify the messages with the trained classifier;

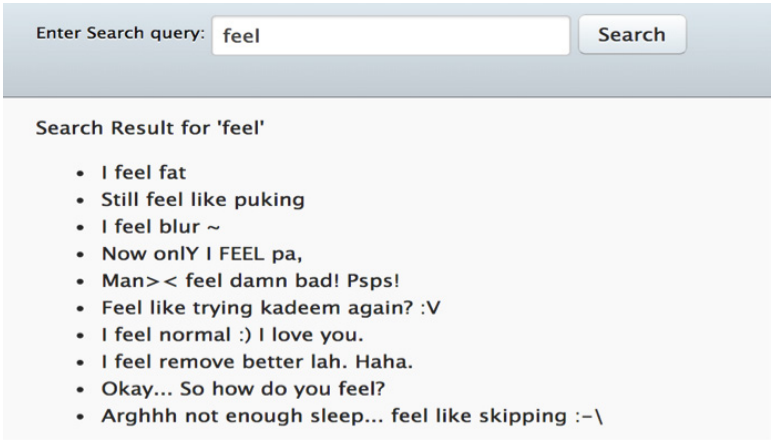


*Figure 2.* Screenshot of the search component.

and (iv) push the results into Solr to enable access by the visualization tool. Note that the techniques for extracting messages from a mobile device are outside the scope of this research. However, interested readers are referred to [2, 3] for details about extracting physical images and SMS messages from Android devices.

The visualization tool provides the following features:

- **Search Interface:** A search tool was implemented to enable users to search for occurrences of any desired term in SMS messages. For example, an analyst may be interested in identifying all the messages that mention the word "feel." The search box shown in Figure 2 can be used to enter a search query; the figure also shows the output with the relevant results. While the search tool is useful when an analyst knows what to look for, it is not very helpful in situations where there is no prior knowledge about the keywords that reveal interesting patterns. To address this problem, a sentiment timeline view (discussed below) was developed to help an analyst discover patterns. Additionally, a tag cloud view was implemented to provide information about the most common keywords in SMS messages.

- **Polarity Distribution View:** This view provides a pie chart that presents the percentage polarity distributions of sent and received messages. Figure 3 displays the polarity distribution of sent messages for a person of interest as seen in the dashboard of the sentiment visualization system.

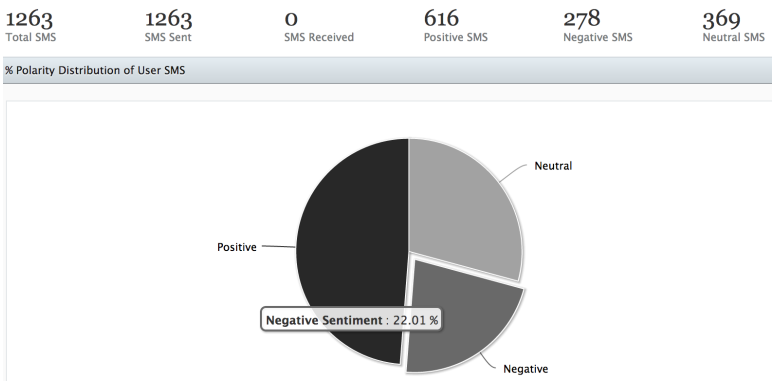


*Figure 3.* Screenshot of the polarity distribution of an individual's SMS messages.

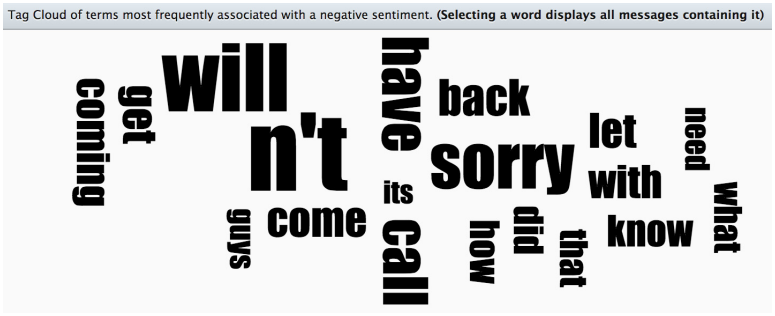


*Figure 4.* Screenshot of the tag cloud of an individual's SMS messages.

- **Tag Cloud View:** A tag cloud is used to render the most common words in messages with negative or positive polarities. This gives an analyst a feel for the terms that are often associated with a specific emotion of an individual. The tag cloud implementation is interactive in that it responds to mouse clicks. When a word in the tag cloud is clicked, SMS messages containing the word are returned. Figure 4 shows a screenshot of the tag cloud generated for a sample individual.

- **Sentiment Timeline View:** A sentiment timeline view (first presented in [3]) was implemented to help analyze the mood swings of an individual over time. Figure 5 shows a screenshot of the timeline view – the horizontal axis represents time and the vertical axis represents the number of messages sent. The sentiment timeline view is at the core of the visualization tool because it provides insights into the emotional swings of an individual in an automated manner.

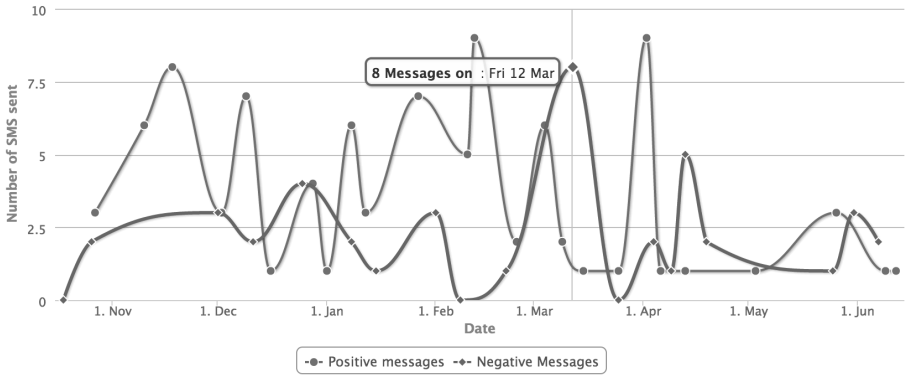


*Figure 5.* Screenshot of the sentiment timeline component.

When the mouse cursor hovers over a node, a tooltip is used to display the number of SMS messages that the node represents. The node may then be clicked to view the contents of the sent messages. As seen in the screenshot, the user experienced a sudden emotional spike on Friday, March 12. This is because the user sent eight negative messages on that day, but did not send any negative messages the previous day. The forensic tool extracts patterns of this nature and reveals emotional fingerprints that would otherwise have been hidden. This feature is more important than a search feature because it reveals insights that a forensic analyst could not acquire via keyword searches. Indeed, sentiment timeline analysis provides very valuable information about the emotionally-volatile periods of a person under investigation.

## 6. Conclusions

This research has attempted to address some of key problems plaguing sentiment analysis in the short message domain. The proposed solution incorporates a sentiment-aware tokenizer, a part-of-speech tagger created for the short message domain, and implementations of normalization and negation. Among all the features considered, part-of-speech tagging proved to be the most effective, followed by stemming. The use of normalization, domain-specific stop words (based on term frequencies) and bigram features absent in previous work further improved the results. Experiments demonstrate that the resulting classifier performs well on an SMS message dataset, validating the similarities existing between SMS messages and tweets, and affirming that the model does not over-fit the data. Additional experimentation with several sentence-level

features demonstrates the utility of normalization in sentiment polarity classification.

A forensic tool was developed to extract sentiment information from short messages sent by persons of interest. The tool also helps visualize the mood swings of subjects over time, assisting forensic analysts in pinpointing periods of emotional instability that may require further investigation.

Future research will focus on the topics discussed in messages. A keyword-based preliminary version of this feature is already provided by the tool in a tag cloud view. Attempts will be made to display topical summaries of a group of messages and correlate these topics with the emotional states of the message sender. To further improve the F-score and classification efficiency, established techniques such as principal component analysis will be used to reduce the feature space. Additionally, receiver operating characteristic analysis will be employed to identify the optimal thresholds for improving classifier accuracy.

## References

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

[2] P. Andriotis, G. Oikonomou and T. Tryfonas, Forensic analysis of wireless networking evidence of Android smartphones, *Proceedings of the IEEE International Workshop on Information Forensics and Security*, 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.), Springer, Heidelberg, Germany, pp. 253–265, 2014.

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

[5] S. Bird, NLTK: The Natural Language Toolkit, *Proceedings of the Association for Computational Linguistics Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*, vol. 1, pp. 63–70, 2002.

[6] 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.

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

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

[9] 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.

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

[11] 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. 841–847, 2004.

[12] A. Go, R. Bhayani and L. Huang, Twitter Sentiment Classification using Distant Supervision, CS224N Final Project Report, Department of Computer Science, Stanford University, Stanford, California, 2009.

[13] B. Han, P. Cook and T. Baldwin, Lexical normalization for social media text, *ACM Transactions on Intelligent Systems and Technology*, vol. 4(1), article no. 5, 2013.

[14] F. Iqbal, H. Binsalleeh, B. Fung and M. Debbabi, Mining writeprints from anonymous e-mails for forensic investigation, *Digital Investigation*, vol. 7(1-2), pp. 56–64, 2010.

[15] X. Jin, Y. Li, T. Mah and J. Tong, Sensitive webpage classification for content advertising, *Proceedings of the First International Workshop on Data Mining and Audience Intelligence for Advertising*, pp. 28–33, 2007.

[16] P. Juola, Authorship attribution, *Foundations and Trends in Information Retrieval*, vol. 1(3), pp. 233–334, 2006.

[17] 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*, vol. 1, pp. 441–448, 2008.

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

[19] E. Martinez-Camara, M. Martin-Valdivia, L. Urena Lopez and A. Montejo-Raez, Sentiment analysis in Twitter, *Natural Language Engineering*, pp. 1–28, 2012.

[20] S. Mohammad, S. Kiritchenko and X. Zhu, NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets, *Proceedings of the Seventh International Workshop on Semantic Evaluation Exercises*, 2013.

[21] O. Owoputi, B. O'Connor, C. Dyer, K. Gimpel, N. Schneider and N. 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 of Computational Linguistics*, pp. 380–390, 2013.

[22] B. Pang and L. Lee, Opinion mining and sentiment analysis, *Foundations and Trends in Information Retrieval*, vol. 2(1-2), pp. 1–135, 2008.

[23] B. Pang, L. Lee and S. Vaithyanathan, Thumbs up? Sentiment classification using machine learning techniques, *Proceedings of the Association for Computational Linguistics Conference on Empirical Methods in Natural Language Processing*, vol. 10, pp. 79–86, 2002.

[24] 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, *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

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

[26] J. Suttles and N. Ide, Distant supervision for emotion classification with discrete binary values, *Proceedings of the Fourteenth International Conference on Computational Linguistics and Intelligent Text Processing*, vol. 2, pp. 121–136, 2013.

[27] P. Turney, Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews, *Proceedings of the Fortieth Annual Meeting of the Association for Computational Linguistics*, pp. 417–424, 2002.

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

[29] H. Wang, D. Can, A. Kazemzadeh, F. Bar and S. Narayanan, A system for real-time Twitter sentiment analysis of the 2012 U.S. presidential election cycle, *Proceedings of the Association for Computational Linguistics 2012 System Demonstrations*, pp. 115–120, 2012.