Computational Linguistics in the Netherlands Journal 2 (2012) 23–33 Submitted 09/2012; Published 12/2012

Choosing the best classifier for the job: Mobile Filtering for the South African Context

Marissa Griesel and **Wildrich Fourie**
Centre for Text Technology (CTexT®), Potchefstroom, South Africa

Abstract

Short messages to cell phones (SMSs) have become the most popular means of communication on digital fronts, especially in Africa and South Africa in particular. This inspires the abuse of such systems by advertisers through the distribution of SPAM. It has therefore become necessary to incorporate a filtering system similar to e-mail classification on these low resource devices. In this article one practical method of filtering such messages for smart phones is described. A prototype application dubbed SpaMiNot intercepts incoming SMS messages and evaluates the message(s) with a number of filters. The application then classifies the message as legitimate (HAM) or unwanted (SPAM). HAM messages are left unhindered, while messages marked as SPAM can either be automatically deleted or stored in a SPAM list, from where it can be restored or deleted. The Android™platform was chosen due to its recent and significant rise in popularity, as well as the ease and speed of Android application development. The application has been thoroughly tested on Android emulators and the Vodafone™845 device.

1. Introduction

The popularity of mobile phones and especially smart phones is increasing at a staggering rate. The number of subscribers in Africa rose from 246 million in 2008 to more than 500 million in 2010 and the preference for so called smart phones over other mobile devices became clear (Rao 2011). Also in this period, 6.1 trillion short messages (SMSs) were sent globally ((ITU) 2010). Naturally, advertisers began exploiting this channel for financial gain. We receive large amounts of text messages each day and not all of these messages are solicited. SPAM (the common name for unsolicited e-mails) is now spreading rapidly to SMSs and this causes mayor increases in the volumes of data that mobile phone users are confronted with. The need for some sort of filtering of SMSs has therefore become apparent, just as e-mail classification has become standard practice. E-mail SPAM filters can, however, not always be used as is for mobile phones, as these devices pose certain restrictions on especially the size of the proposed solution.

This research aimed at investigating the usability of algorithms traditionally used for e-mail classification for SMS filtering and selecting the best solution for the particular context. The chosen method was then applied in developing a prototype SMS SPAM filter for Android™[1] operated smart phones that can minimise some of the irritation caused by these unsolicited messages for South African users. The resulting application was affectionately named SpaMiNot and initial user feedback is very positive.

In this article, the relevant concepts taken from e-mail classification and other related work will first be discussed, followed by a description of the South African SMS environment that needed to be taken into consideration during the design of the application. The surface filtering, making up the first of a two-part architecture, will then be discussed before the more complex content filtering comes under the loop. Four conventional algorithms commonly used in e-mail content filtering were applied and their effectiveness compared to a fifth classification method that uses a simplified route, adapted to the South African context. The statistical significance of the results were computed and

1. For more information and to download the operating system, see `http://www.android.com/`.

the classifier with the significantly greatest improvement incorporated in a real-world SPAM filter. The evaluation of the complete system is given before the article concludes with a list of future work.

2. Related Work

Unsolicited SPAM e-mail is a well known, researched and understood problem. Symantec's annual MessageLabs Intelligence security report Symantec (2010) indicated that 89 % of all e-mails sent can be classified as SPAM. It is therefore not surprising that the usefulness of a wide range of techniques to classify e-mail as SPAM (unsolicited and unwanted messages) or HAM (messages containing solicited or useful information) has been researched (see for instance Sebastiani (2002), Dwork et al. (2003), Hall (1998), Golbeck and Hendler (2004), and Tompkins and Handley (2003).) The most popular technique when classifying e-mail, has proven to be machine learning. In machine learning, features inherently included in SPAM messages are extracted and represented in a number of algorithms that yield good results (see for instance Androutsopoulos et al. (2000)).

In the past few years, the adaptability of these techniques to the mobile market has also received some attention, although this research mostly looks at using existing methods for e-mail classification, rather than creating domain specific methods. In Hidalgo et al. (2006), different machine learning algorithms were tested for the mobile domain. This study used a corpus of 82 SPAM messages and 1119 HAM messages (all in English), collected from especially university students. These messages were then classified using an adapted version of Naive Bayes, C4.5, PART and Support Vector Machines. The best results over these four algorithms were achieved using SVMs and ranged from 89% to 95%, depending on the number of features and the certainty level of the classifier. It is interesting to note that the C4.5 classifier never performed on par with the other models and was left out for advanced experimenting. Not all machine learning algorithms therefore show the same level of adaptability to SMS filtering.

The work in Junaid and Farooq (2011) was based on a much larger corpus of 2000 SPAM messages and 6000 HAM SMSs, also only in English. Algorithms tested here included Naive Bayes, K-nearest neighbour, C4.5, JRip and Support Vector Machines, as well as Fuzzy AdaBoost, GAssist-ADI and XCS. Results ranged from 50% accuracy over 3000 unseen SMSs to 78% for Naive Bayes classification.

Our own work in the domain of SMS classification is limited to research for development of SpaMiNot and, to our knowledge, it is the only research done thus far for any of the South African languages. This paper therefore serves as an important follow-up to the initial experiments as reported in Griesel and Fourie (2011) and focuses on optimisation of the content filter (section 5).

3. The South African Environment and Specific Design Aspects

Before development of SpaMiNot could commence, the specific factors that would have an impact on the design of the SMS filter had to be determined. The biggest consideration for the mobile market, is probably the size of the application. The application should be able to function on the mobile device without impeding the normal day-to-day operation and without disturbing other critical functions. Also, it was deemed more important to receive all HAM messages than to filter out all SPAM messages as valid HAM might contain important information that the user cannot afford to lose. These and other limitations lead to important design decisions.

Typical e-mail SPAM filters are deployed on large mail servers which have adequate capacity for classifying a message with resource intensive algorithms. This makes classification of an e-mail as SPAM or HAMS a very accurate domain. Preliminary talks with representatives of the South African mobile network operators have indicated that the costs of incorporating a filtering system on the network servers far outweigh the profit made by the SMS service. The possibility of creating a similar server-based filtering system is therefore not a viable one in South Africa. An SMS filtering

system would have to be entirely dependent on the resources of the mobile phone on which it is installed. The average mobile processors have very limited resources available and the application should therefore be as small (a size limit of 2 MB was decided on) and computationally inexpensive as possible. The Android mobile platform (version 2.2) was identified as the operating system which would allow the application to function on the widest range of devices.

Corpus collection was an extremely difficult task as almost all approached parties declined to provide any of their SMSs, even the SPAM messages. Privacy was the key deterrent even though the option was given to anonymise the data. The final development corpus (a first of its kind for South African languages) consisted of 728 HAM SMSs and 329 SPAM messages with equal amounts of Afrikaans and South African English words. Most HAM messages contained a mixture of English, Afrikaans and "SMS-shorthand" words and were mostly meant as informal communications so the senders did not adhere to the standard variant of either predominant language. The only criteria for these messages were that it had to be written and received by South Africans and that the language should be either SA English or Afrikaans.

The decision to include two languages in the corpus simultaneously was made more out of necessity than creativity, since most SPAM messages in South Africa are written in English. In our corpus of just over 300 messages, all but one was in English. The only example of Afrikaans SPAM was from a publishing firm, specialising in Afrikaans poetry and prose. To make sure that SpaMiNot was not in effect an elaborate language identifier, the HAM messages were in both English and Afrikaans. Most of the messages were obtained from university personnel, students and friends, as well as from the Wireless Application Service Providers' Association (WASPA)[2].

To optimise classification accuracy and minimise computational intensity, it was decided to utilise a set of filters that could be applied sequentially. The filters each handle one aspect of a typical South African SMS and aim at classifying the message as SPAM or HAM as soon as possible. This cascading method leaves the computationally more complex filtering as a last resort and first tries classification on the basis of the less complex filters. The user can also disable or enable any of the filters, creating a customised configuration of active filters according to his/her needs. One of the most important settings asks the user to decide whether an SMS classified as SPAM should be deleted, or simply moved to a designated folder on the device. This ensures that users who regularly receive important information via SMS will still receive a notification that the message was intercepted and moved and can use their own discretion to decide which to delete.

A graphic representation of the implementation of SpaMiNot can be seen in Figure 1. The next two sections describe the various filters - surface filtering and the five methods tested to filter the content of the SMS messages.

4. Surface filtering

After a close analysis of the SMS corpus, a number of filters based on the surface characteristics of the corpus were implemented. These filters aim at making the application faster and less resource intensive, as they are applied before the costly content filtering described below. Incoming messages are therefore first scrutinised on the basis of their inherent qualities before the content of the message is parsed and used to classify the message.

1. Lists Filter: Untrusted numbers (of known distributors of SPAM) can be stored in a blacklist while trusted numbers can be stored in a whitelist. Messages received from blacklisted numbers are automatically tagged as SPAM, while messages received from whitelisted numbers are automatically tagged as HAM. The black- and whitelists are not pre-populated as the user should customise this feature depending his/her needs. In the case where the source of the

2. This South African organisation encourages users to lodge complaints against companies that misuse mobile technologies for financial gain. More information is available at `http://www.waspa.org.za/index.shtml`.

current scrutinised message could not be found in either the white- or black-lists the next enabled filter, as depicted in Figure 1, will be utilised to attempt to classify the message.

2. Contacts Filter: This filter only allows messages from known contacts stored in the contact list (phonebook) of the user's mobile phone. Any message sent from an unknown user will automatically be flagged as SPAM and handled accordingly.

3. Long Number Filter: A common practise of SPAM'ers is to send messages from the Internet. These messages originate from numbers that are longer than the South African maximum of 12 digits. A typical message that was sent from the internet will for example show that it originates from a number like +27 82 555 5555 55555, while a message originating from a mobile phone will show that it originates from numbers such as +27 83 555 5555 or 083 555 5555. An option could be implemented to allow the user to customise the length requirement of the long number filter in order to adapt the application for international users.

The effectiveness of these filters lie in the simplicity thereof. A message can easily be categorised and handled appropriately without performing any complex manipulation. Even without processing anything other than the number from which the SMS originated, 32% of messages could be correctly classified during evaluation (see section 6). This also results in gains on size and speed. Figure 1 gives an overall view of the application of the different filters.

5. Content filtering

The last and most computationally complex filter utilises the text content of the SMS to classify a message as SPAM or HAM and this is also the section that has the most potential for inclusion of different natural language processing techniques. As machine learning proved to be the most effective classification method in related international studies, the popular algorithms in this field were first tested for the specific context. The training with 4 different algorithms, as well as a 5^{th} method is described below.

5.1 Machine Learning

To establish the best method of content filtering for this proposed application, four machine learning algorithms widely used in e-mail classification tasks were applied to the development (training) corpus. The four algorithms showed promising results in other international research described earlier and include Decision Trees, Naive Bayes classification, Support Vector Machines and Multinomial Naive Bayes classification. The WEKA data mining toolkit (Bouckaert et al. 2010) with a default experimental setup was used to facilitate training.

All four algorithms were used to train a binary classification model (HAM vs. SPAM) from the collected corpus. Each SMS represented one training instance and to further reduce the amount of processing needed at run-time on the mobile device, the features were limited to a string vector of all the words in an SMS message, which maps words to term occurrence frequencies. The four resulting models were evaluated on the development corpus using 10-fold cross validation. These results, as well as the size of the resulting classification model are given and discussed in Table 2 at the end of section 6.

5.2 An alternative method

Machine learning algorithms are very dependent on the size of the training corpus and since we did not have a very big collection, it was decided to also test a simplified approach. The basic premise, under which this filter operates, is that certain key phrases like "SMS STOP TO OPT OUT", "GREAT SAVINGS" and "SMS TO WIN" frequently occur in SPAM messages, but not in HAM.

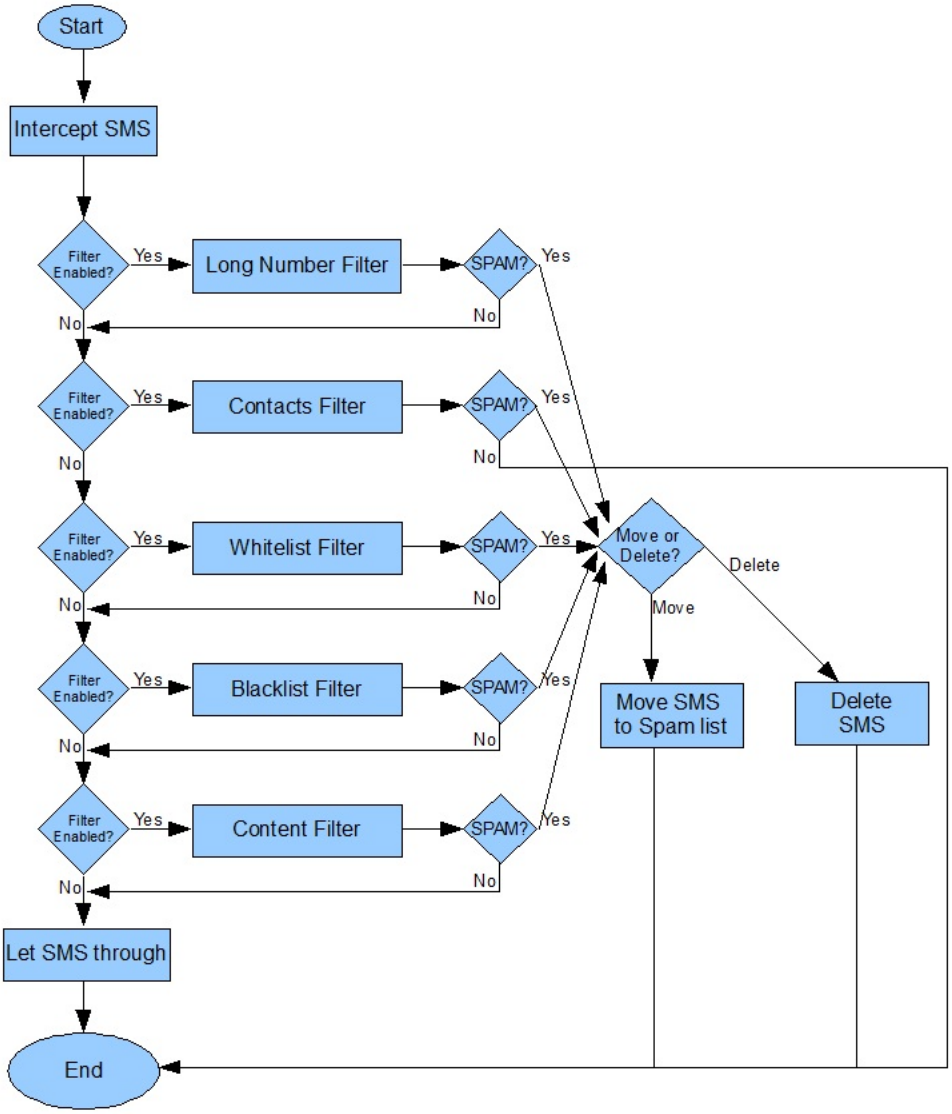

Figure 1: Two stage architecture of SpaMiNot

Table 1: Message classification success of the different threshold values

Threshold Value	**SPAM % correct**	**HAM % correct**	**Precision**	**Recall**	**F-measure**
-50	98.689	77.503	0.775	0.992	0.870
-40	98.689	80.137	0.801	0.993	0.887
-30	98.253	82.286	0.823	0.990	0.899
-20	96.943	85.164	0.852	0.984	0.913
-15	96.506	86.582	0.866	0.982	0.920
-10	96.069	88.163	0.882	0.980	0.928
-5	**95.196**	**98.581**	**0.992**	**0.980**	**0.986**
0	**94.759**	**99.229**	**0.994**	**0.977**	**0.985**
5	92.576	99.554	0.996	0.967	0.981
10	90.393	99.635	0.996	0.958	0.977
15	89.082	99.635	0.996	0.953	0.974
20	85.589	99.756	0.998	0.939	0.967
30	80.349	99.797	0.998	0.918	0.956
40	77.729	99.837	0.998	0.908	0.951
50	74.672	99.837	0.998	0.897	0.945

The word choices used in SPAM messages are therefore a distinguishing factor when classifying a message.

A bag-of-words approach was followed to construct frequency lists from the SPAM and HAM messages. An index score for each word was then calculated using the following formula:

$$index = \begin{cases} \frac{SPAM\ Frequency}{HAM\ Frequency}, & SPAM\ Frequency \leq HAM\ Frequency \\ \left(\frac{HAM\ Frequency}{SPAM\ Frequency}\right) \times -1\ , & SPAM\ Frequency\ > HAM\ Frequency \end{cases} \quad (1)$$

The index of a particular word is essentially the ratio in which the word occurs in SPAM and HAM messages with words occurring more frequently in HAM obtaining a negative value. To classify a message it is scored by totalling all of the index values of the words contained within the message. This gives a unique threshold value for each incoming message. In the corpus collected for this research, SPAM messages are generally 51% longer than HAM messages. By not averaging the total this factor is also taken into account. Through efficient programming the additional amount of memory used to classify a message would depend on the size of the word list used.

The complete word-index list (the list of words each accompanied by their index value) contained 3406 words and took 37 kilobytes of space which is adequately small to keep memory usage low. The list is stored in a SQLite transactional database on the mobile device, which allows for the best scalability of persistent data. To find the optimal value which divides the HAM and SPAM message scores, a range of threshold values were tested. Message scores for each message in the corpus were calculated using the index values in the word-index list. A message with a score equal or above the threshold was classified as SPAM and a score below the threshold was classified as HAM. The percentage of correct classifications for each threshold was then calculated and is represented in Table 1. It should be noted that these percentages take false negatives and false positives into account.

When deciding which threshold to incorporate into SpaMiNot's final design, the level at which the user would experience the least amount of frustration was taken into account. It is important that false positives (messages incorrectly classified as SPAM) were not presented as HAM too often, as this would deem SpaMiNot an unnecessary waste of valuable resources on any mobile device. Equally important, however, is not to classify too many HAM messages as SPAM (false negatives),

as this would irritate the user when they do not receive valid information. From the experiments, the threshold values of 0 and -5 gave the best results, with the former achieving the objective for the best classification of HAM messages, although the latter threshold classified more of the SPAM messages correctly. Both the differences are quite minimal for this limited evaluation set and the difference might be greater with larger samples of corpora. All things considered, a threshold value of -5 was incorporated. Note that a threshold value of zero indicates that the message contains an equal amount of words typically found in SPAM and HAM. The evaluation is further discussed in the next section.

6. Evaluation and Discussion

The WEKA toolkit also contains an automatic evaluation module. The four models trained in this environment were scored using 10-fold cross validation. Although this evaluation method is not perfect for evaluation of an application that will be used in varying circumstances and with a broad user profile, the already small size of the development corpus did not grant the luxury to use a separate evaluation corpus. A very small subset was already extracted as final test set to evaluate the system in its entirety - incorporating surface and content filtering.

The F-measure is the harmonic mean between the precision and recall and indicates a weighted measure of a test's success with equal weights assigned to the precision and the recall values (see Van Rijsbergen (1979)). It is calculated using the following formula:

$$F - measure = \ 2 \times \left(\frac{precision \ \times recall}{precision + recall} \right) \tag{2}$$

From Table 2 it is clear that the machine learning algorithms indeed suffered because of the small training set. All four of these methods delivered good results, but did not satisfy the requirements set out at the beginning of the experiment. Of the machine learning algorithms, the Support Vector Machines delivered the best results, but still classified 14 of the HAM messages as SPAM and it includes the biggest classification model. In the real-world situation for which this application is designed, a user would simply uninstall it to make space for a more useful application as the frustration of not receiving valid information might outweigh the benefits. Looking ahead, the team also saw the possibility of incorporating some sort of active learning to customise the content filter to the individual user. This would entail retraining any machine learning model either on the mobile device itself, or by connecting the device to a computer. Although this possibility is not at all farfetched, retraining of the bag-of-words approach seems a simpler process and one that could be performed more often on the device itself. This would make SpaMiNot more user friendly and effective.

The bag-of-words design ultimately incorporated in SpaMiNot not only delivers a much smaller application to be installed on the mobile device, but it also outperforms the four conventional models with regard to accuracy. The evaluation of the content filter in isolation when tested on the development corpus (using 10-fold cross validation), indicated that a threshold value of -5 correctly classified the most messages with a success rate of 95.2% for SPAM and 98.6% for HAM.

An analysis of variance was computed on each of the content filtering approaches using the F-measure since it represents the harmonic mean between the precision and recall. Our null hypothesis is that all of the different approaches have the same mean F-measure value. The results of this experiment are represented in Table 3, and based on the results the null hypothesis was rejected.

We then compared the different means of each of the machine learning approaches with the mean of the bag-of-words approach using a post-hoc Tukey test. The results indicated that the differences between the bag-of-words approach and the J48, Naive Bayes and the Multinomial Naive Bayes approaches were statistically significant. The difference between the Support Vector Machines and bag-of-words approaches were not statistically significant, indicating that the Support Vector Machines could have been an effective approach if the model did not take up the most space (see

Table 2: Comparison of content filtering approaches

Algorithm	**Success**	**Success**	**F-measure**	**Size**
Algorithm	**SPAM**	**HAM**		**(in KB)**
Bag-of-words	95.2%	98.6%	**97.0%**	**37**
Decision Trees (J48)	84.4%	93.3%	**90.5%**	**337**
Naive Bayes	89.1%	94.5%	**92.5%**	**585**
Support Vector Machines (SMO)	93.3%	97.1%	**95.9%**	**809**
Multinomial Naive Bayes	90.4%	95.4%	**93.8%**	**355**

Table 3: Analysis of variance on the F-measures

Source of Variation	**SS**	**df**	**MS**	**F**	**P-value**	**F crit**
Between Groups	0.028	4	0.007	17.277	<0.001	2.579
Within Groups	0.018	45	0.000			
Total	0.046	49				

Table 2). This just goes to prove that the mobile context indeed has a very big influence on the choice of solution.

For evaluation of the application as a whole, incorporating the entire set of filters (surface and content filtering), an additional 20 HAM and SPAM messages were collected to act as the test set. These SMSs were not contained in the development corpus and were classified using the full SpaMiNot setup. Each SMS was first measured against the three surface filters. Of the 20 instances, 4 were correctly classified at this early stage and only 16 SMSs were presented to the content filter. Here, only one SPAM message was incorrectly classified as HAM and the application achieved an overall accuracy of 95%. This test set is relatively small, but in a data scarce environment like this, the development team did not want to leave out too many messages during training. More important for user satisfaction is that SpaMiNot misclassified no HAM messages in this experiment.

To further gage the satisfaction of users with the application, it was given to a group of smart phone users. This test group of 10 independent volunteers were asked to install and activate SpaMiNot for one week at the end of the month (as it was clear from our corpus that spammers were much more active then). At the end of the week, all the group members had to vote in one of three ways - would they a.) immediately deactivate SpaMiNot, b.) keep using the application indefinitely, or c.) are not sure yet. 7 of the volunteers voted for B, 1 each for A and C respectively and one volunteer did not vote.

7. Graphic User Interface

In the competitive mobile market, with application stores (or "app stores") spreading huge amounts of downloadable content for smart phones at relatively low costs, it is just as important that SpaMiNot has a user friendly interface. A simple design was chosen to enable the user to quickly and easily customise the different filters (activating or deactivating the appropriate surface filters and populating the black- and whitelists).

Figure 2 gives an example of a received SPAM message and the main menu. The user also has the choice of deleting the messages classified as SPAM, or simply moving these to a designated folder in the file management system of the mobile device. Figure 3 shows the information message that appears when a new SMS is received and the menu to move or delete an SMS.

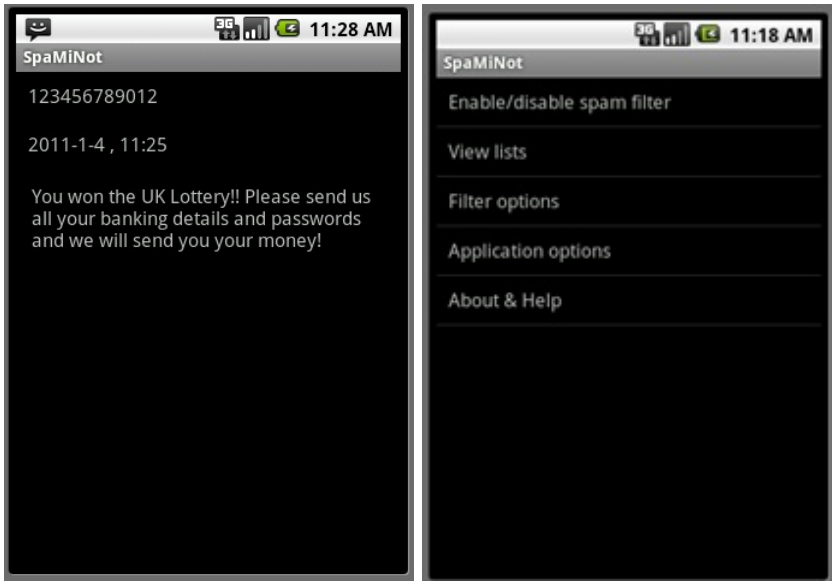

Figure 2: Example of a SPAM message and the main menu of SpaMiNot.

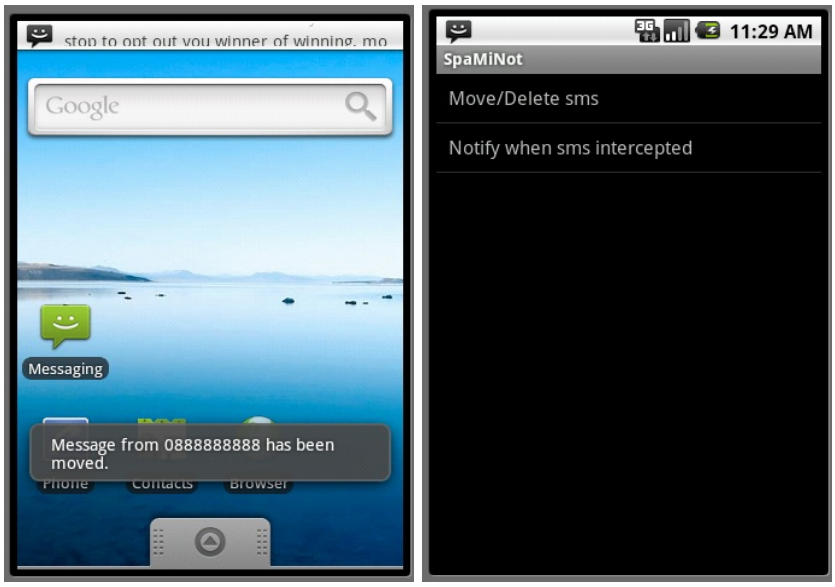

Figure 3: An intercepted message and the move or delete menu of SpaMiNot.

8. Conclusion and Future work

With this prototype application, we have developed a framework for future work. The cascaded filtering system is currently optimised for an English or Afrikaans user as all of the data contained in the (very small) corpus originated from first language speakers of one of these two languages. More research into optimisation for other languages is necessary and future work will depend heavily on the collection and availability of data. We would especially like to investigate these techniques in language specific corpora for the South African languages.

One particular area of interest for future development includes the reconstruction of the word list and recalculation of the index values with the addition of words contained in newly received messages on the mobile device itself. This technique would make the portability of the technique to other languages and the inclusion of user specific words used in HAM messages a more automated process with SpaMiNot learning from its user and adapting accordingly. The SPAM word index would also benefit from such recalculation, as spammers are also evolving to undermine filtering efforts. As their "sales lingo" changes, SpaMiNot will be able to adapt to these changes and stay as effective. Portability to other smart phone operating systems is also an area of future work.

Other filtering techniques and combinations of these filters should also be investigated. From the small amount of available data, it has become clear that spell checking might be a valuable feature. The SPAM messages in the corpus were usually spelled more correctly than HAM messages that included a large amount of "SMS-shorthand", so using the amount of words flagged as spelling errors in a message might serve as a valuable classification feature. One consideration here would be the size of the spelling checker itself and implementation should be considered carefully. The amount of punctuation in an SMS might also be a distinguishing factor as SPAM contained on average 4.6 punctuation characters per message, compared to 3.2 punctuation characters per HAM message. Another interesting observation is that SPAM contains on average 25% more exclamation marks than HAM.

SpaMiNot is an ideal first step into the emerging field of SMS filtering with very limited resources in the South African environment. As the market for mobile devices and the rapid pace at which these platforms are exploited grows, so too do the possibilities for natural language processing to make it a safe and well-managed environment for all users.

Acknowledgments

The authors would like to thank the Centre for Text Technology (CTexT®) of the North-West University for financial and other support, Menno van Zaanen (University of Tilburg) together with all participants in CLIN 22 for comments and ideas regarding certain design aspects as well as all contributors to the SMS corpus.

This material is based upon work supported by the South African National Research Foundation. Any opinion, findings and conclusions or recommendations expressed in this material are those of the author(s) and therefore the NRF does not accept any liability in regard thereto.

References

Androutsopoulos, I., J. Koutsias, K.V. Chandrinos, G. Paliouras, and C.D. Spyropoulos (2000), An evaluation of naive bayesian anti-spam filtering, *Prceedings of the Workshop on Machine Learning in the New Information Age, 11th European Conference on Machine Learning (ECML 2000)*, pp. 9–17.

Bouckaert, R.R., E. Frank, M.A. Hall, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten (2010), Weka-experiences with a java open-source project, *Journal of Machine Learning Research*, Vol. 11, pp. 2533–2541.

Dwork, C., A. Goldberg, and M. Naor (2003), On memory-bound functions for fighting spam, *Proceedings of the 23rd Annual International Cryptology Conference (CRYPTO 2003)*.

Golbeck, J. and J. Hendler (2004), Reputation network analysis for email filtering, *Proceedings of the First Conference on Email and Anti-Spam (CEAS)*.

Griesel, M. and W. Fourie (2011), Mobile content filtering for the south african market, *Proceedings of the Twenty Second Symposium of the Pattern Recognition Association of South Africa (PRASA)*.

Hall, R.J. (1998), How to avoid unwanted email, *Communications of the ACM*.

Hidalgo, J.M., G.C. Bringas, and E.P. Sanz (2006), Content based sms spam filtering, *Proceedings of the ACM symposium on Document engineering*, pp. 107–114.

(ITU), International Telecommunications Union (2010), The world in 2010: Ict facts and figures, *Technical report*, International Telecommunications Union (ITU). [Accessed on 12-09-2011]. http://www.itu.int/ITU-D/ict/material/FactsFigures2010.pdf.

Junaid, M.B. and M. Farooq (2011), Using evolutionary learning classifiers to do mobile spam (sms) filtering, *Proceedings of the ACM symposium on Document Engineering*, pp. 153–161.

Rao, M. (2011), Mobile africa report 2011: Regional hubs of excellence and innovation, *Technical report*, MobileMonday. [Accessed on 12-09-2011]. http://www.mobilemonday.net/reports/MobileAfrica_2011.pdf.

Sebastiani, F. (2002), Machine learning in automated text categorization, *In ACM Computing Surveys,*, Vol. 34(1), pp. 1–47.

Symantec (2010), Messagelabs intelligence: Annual security report, *Technical report*, Symantec. [Accessed on 12-09-2011]. http://messagelabs.ch/mlireport/MessageLabsIntelligence_2010_Annual_Report_FINAL.pdf.

Tompkins, T. and D. Handley (2003), Giving e-mail back to the users: Using digital signatures to solve the spam problem, *First Monday*, Vol. 8(9).

Van Rijsbergen, C.J. (1979), *Information Retrieval*, Butterworth-Heinemann. London. ISBN: 0408709294 (224 pages).