A text-mining approach to helpdesk and e-mail support

A. N. Blaafladt B. H. R. Johansen N. E. Eide F. E. Sandnes

Faculty of Engineering
Oslo University College

Abstract

The system administrators of large organizations often receive a large number of e-mails from its users and a substantial amount of effort is devoted to reading and responding to these e-mails. The content of the messages can range from trivial technical questions to complex problem reports. Often these queries can be classified into specific categories, for example reports of a file-system that is full or requests to change the toner in a particular printer. In this project we have experimented with text-mining techniques and developed a tool for automatically classifying user e-mail queries in real-time and pseudo-automatically responding to these requests. Our experimental evaluations suggest that one cannot completely rely on a totally automatic tool for sorting and responding to incoming e-mail. However, it can be a resource saving compliment to an existing toolset that can increase the support efficiency and quality.

1 User dynamics: Problem statement

Reading and responding to e-mails from disgruntled users in an organization takes up several hours of a system administrator's daily effort. At the Engineering faculty at Oslo University College, with some 1,500 users, system administrators often encounter an inbox with hundreds of messages in the morning when coming to work. The task of responding to these e-mail messages can be daunting, time-consuming and tedious. Yet, timely and quality replies are immensely important for the individual user in order for the users to fulfill their role in the organization. Technical difficulties and setbacks, that may seem trivial to a system administrator, can be overwhelmingly frustrating and destructive for the user and can be unnecessarily costly for the organization. Further, it is important that anomaly reports from users get the attention of the system administrators early so that corrective and preventive actions can be taken and such that the damage and the repercussions of the anomalies are minimized. It is therefore important that the system administrators. inboxes are continuously monitored.

Until very recently there have been few formal training programs for system administrators with the exception of certification for brand-name equipment etc. Yet, the existing training and certification programmes primarily focus on technical issues. User support of usually learnt on the job and there are often few official procedures for how to handle user queries. Attempts at ISO-certifying and standardizing procedures in organizations may result in some of these queries being formalized, but often it is superficial bureaucracy that, in terms of timeliness and responsiveness, may actually

reduce quality rather than improving it. Consequently, system administrators often develop their individual practices for handling user queries, and often these "common sense" practices work quite well in practice.

Organizations are dependent on a team of system administrators and these administrators can usually be reached through one point-of-call, such as support@fantasticCorp.com. E-mails sent to such an address are commonly forwarded to the entire team of system administrators that are responsible for user queries. Two obvious reasons behind this strategy are that it is easier for the users to remember and the administrator on-duty will receive the message. However, often individuals in an organization know one or more of the system administrators personally and may decide to send a message to this specific system administrator directly. This is for the reasons stated an undesirable practice.

The size of the inbox is dependent on a number of factors. Two important factors are when the inbox was last inspected and the occurrence of certain system events. According to queuing theory e-mails can be modeled as arriving in a stream with a Poisson distribution, and therefore the size of the inbox is approximately proportional to the time passed since it was last read. This phenomenon is something that everyone coming to work in the morning or returning from lunch, meeting or even a business trip can testify. Further, the occurrence of certain events, such as a system failure or anomaly usually ignites a burst of e-mails. For example a printer breakdown on Friday afternoon does not go down well with users struggling to meet their deadlines. Once a system failure or anomaly affects the work carried out by the user the users often chose to resolve the problem by contacting the system administrators by e-mail, as it may be hard to reach the system administrator by phone. The more users an anomaly affect the more e-mails the system administrator receives. Occurrences of anomalous events are hard to predict but they themselves may be modeled suing a Poisson-like distribution.

The messages from users can be coarsely classified into four categories: a) automatically generated mails from various processes such as cfengine, at, cron etc, b) unsolicited e-mail and spam, c) general questions and d) urgent questions, notifications and anomaly reports. Automatically generated mail is easily sorted into assigned folders using simple keyword e-mail filters, and automatic spam filters are getting increasingly good at reducing the amount of junk in the inbox. General questions from users, such as how to do "this or that", are often not urgent and the task of responding to the question can be delayed to a "low-peak time". Often, users ask similar questions and the system administrator can retrieve an old answer written to a different user at a previous point in time. In this way, the administrator hopefully saves time if this archived message can be recalled with little effort. However, urgent e-mails from users reporting anomalies, faults and strange behaviour in the computer system should be read and dealt with immediately. Many system administrators in small and medium sized organization get a "feeling" for how to read and respond to e-mail. The purpose of this work is to assist the system administrator by simplifying e-mail management. DIGIMIMIR, based on text-mining techniques, automatically clusters and categorizes incoming e-mail into related topics and presents the e-mail categories in terms of indentifiable and characteristic keywords. System administrators gets a clear overview of the inbox contents, and can thus more easily identify urgent matters that needs to be resolved at high priority. In addition, DIGIMIMIR can be connected to a reservoir of pre-answered questions such that the most suitable answers to commonly asked questions is found automatically.

2 Previous work

Much has been written on helpdesk support [24], [10], [7], [12] and many commercial systems exist, for example E11 and IssueTracker. These systems primarily focus on tracking historic aspects of customer support, maintenance of searchable knowledge bases and the identification of reoccurring issues. Many of these products target general businesses. GNATS is a system specifically designed for tracking bugs in software and the maintenance of these in databases [23].

Trouble ticketing systems, such as OTRS (Open Ticket Request System), are useful tools for managing large inboxes, which may be handled by several system administrators concurrently. New requests that arrive in the inbox are given a ticket (e.g. a number) and an automatic reply informing the user that the request is received and will be handled. Other requests on the same issue are automatically grouped together with the existing correspondence related to the ticket. The different system administrators therefore have easy access to the entire history of the requests associated with a ticket. The responses of different system administrators can therefore be coordinated and uniform.

Expert system based helpdesk systems have also been explored [30], where the staff running the helpdesk is guided through sequence of decision rules to solve the particular difficulties. The problem with expert systems is the nontrivial establishment of the decision rules. Another strategy is case based reasoning which is especially suited at detecting recurring problems [3]. In the spoken domain recurrent neural networks have been used to route helpdesk calls automatically based on utterances [8].

The difficulties of handling large number of requests are commonplace in large organizations. Research effort has gone into the automatic retrieval of answers from existing question-answer lists based on queries, for example the VIENA classroom system, which uses lexical similarity [27], [28], the FAQFinder system [1], [17] which uses semantic knowledge, the FAQIQ system which uses case based reasoning [18]. In a different approach [25] a template strategy is used to answer questions based on information in relational databases. Common to all these strategies is that they are based on already existing knowledge bases.

In addition to the distribution of answers it is also necessary to categorize questions. The idea of automatically sorting e-mail into predetermined categories has been examined by several researchers. In one theoretical study [31] web-mining agents are assessed as a means of automatically sorting e-mail using an uncertainty probability based sampling classifier and rough relation learning.

Recently, there has been a huge public interest in the problems associated with spam, and a substantial effort has gone into developing spam-filtering technology [1], [29]. Notably, by far the most efficient strategy is the statistically based naïve Bayesian classification [29]. A naïve Bayesian system is trained using a large corpus of spam e-mails and non-spam e-mails and a word signature vector is established for both groups. When a new message arrives, its word signature is compared to both that of the spam and the non-spam signatures and the one that yields the best match determines its category. The spam-filtering problem is related to the problem we are addressing in this paper. However, it is different on two major accounts. Firstly, spam-filtering entails classifying messages into two groups, either spam or non-spam. Second, these categories are defined a-priori. However, in document classification there are many (usually more than two) categories and the categories might not be known and therefore have to be established at run-time. Bayesian classification has in fact also been studied in the context of general document classification [20]. One exciting practical development that has occurred in

parallel with the development of DIGIMIMIR is POPFILE, which is a Bayesian based e-mail classification program [11]. POPFILE works directly on POP3 e-mail accounts and uses Bayesian classification to sort incoming messages into predetermined categories. POPFILE as a software product has reached some maturity, but still suffer from major shortcomings . the most notable (at time of writing) is the lack of IMAP support. The IMAP protocol is particularly applicable in the context of automatic mail sorting applications since mail folders can be managed on the server [4], and it is therefore possible to achieve e-mail client independence. Further, POPFILE is designed only to work in supervised mode and is reliant on training data and therefore cannot be deployed in unsupervised situations where new categories are created dynamically on the fly.

Another exciting and controversial new development is the announcement of Google's new e-mail service Gmail [9]. This is a novel strategy in terms of managing e-mail. The idea is that the users get a large area to store their mails and that these e-mails therefore rarely have to deleted. Document classification techniques are therefore used to navigate and search through this huge reservoir of e-mails. This service is not yet available to the general public, but this new thinking with regards to dealing with e-mail management may prove to be useful in terms of support and helpdesk e-mail management.

In fact, the largest success of text mining and document classification and retrieval technologies has been within the areas of web search engines and search engine technology has advanced a rapid pace of the last few years. However, there is a number of profound differences between the clustering and classification requirements for indexing web pages and handling streams of incoming e-mail. First, the largest difference is probably the volume of text. The web is huge and the size can be exploited to up the quality of the clustering and classification. Personal e-mail collections or an organization's e-mail collection remain small by comparison. Second, indexing of web pages can be done offline and there are no critical time-restrictions on how fast the pages are indexed. Indexing is often done offline and there is a significant turnaround time from when a change is being made on a document on the Internet until it is being reflected in the search results of the search engines. We could perhaps describe this as an index epoch. However, in order for an e-mail sorting system to have a value the classification and clustering needs to be continuous, instant and real-time. Third, in the indexing of web pages quality is the prime importance, while in the clustering and classification of e-mail messages quality can be traded for speed. Until now, most of the research into text mining and document classification and retrieval primarily focus on clustering and classification quality and they pay less attention to operational issues such as real-time constraints incrementally growing document collections (inboxes).

Finally, some system administrators wisely believe that the best support policy is self-reliance and that users should be able to resolve their own problems as much as possible [6] and hence reduce the need for support.

3 DIGIMIMIR and text-mining

DIGIMIMIR employs techniques borrowed from web mining (see [2] for a general introduction to web mining) and terminology mining based on text corpora (see for example [14]). DIGIMIMIR takes a set of messages as input and produces a set of message clusters as output, i.e. related messages are clustered together, and unrelated messages are placed in different clusters. The step comprises several phases. First, the system must retrieve the messages, then each message is pre-processed and transformed into a text vector. The set of text vectors representing the set of messages are used as input

to the clustering algorithm for then to compute the most suitable clusters and finally the results are presented to the recipient (user).

A dedicated e-mail account is set up and DIGIMIMIR polls the inbox at regular intervals to check for new incoming messages. The inbound messages are processed as follows: Each message is treated as a separate entity and used as a basis for computing a word vector. A word vector represents a set of words as a vector, where each word in a dictionary is assigned a specific position in the vector, thus the size of the vector equals the size of the dictionary used. The presence of a word is marked by a positive non-zero integer, where the value represents the number of times the word occurs in the text. A zero denotes the absence of a specific word. Cleary, different messages containing different words have different word vectors in the word space. Hence the phrases “nuts and bolts” and “bolts and nuts” would both yield the same word vector.

To generate a word vector the text is organized into individual words. The first step is to filter high frequency words, also known as stop words [26], [13]. This is achieved using a stop word dictionary computed from word frequency lists [15]. Next, word stemming is employed to obtain the general form of words [22] with the purpose to reduce the size of the word vectors. Then, a dictionary-based spell checking technique is used. I.e. a reference dictionary comprising of all the possible valid words in the language in all grammatical forms is used. If an entry in the text dictionary cannot be found in the reference wordlist then the entry is tagged as a potential incorrect spelling. All entries marked as potential incorrectly spelled words are crosschecked against the reference wordlist using Metaphone [21]. Metaphone is a technique, inspired by SOUNDEX for matching words based on their English phonetic sound and it is particularly suitable for spell checking applications. Entries with no Metaphone match in the dictionary are considered special terms. Entries with a metaphone match are most likely incorrectly spelled words (see [16] for an excellent survey of automatic spelling correction techniques). Then, each instance of the remaining words is counted and represented in the word vector. The word vector therefore represents the potentially interesting words that are characteristic of the question [5]. Further, the word vectors can be large and contain large amounts of noise. Research into text-mining often strives to reduce the dimensionality, or size, of the word vectors by the means of some transformation [29]. In DIGIMIMIR a simple word vector reduction technique is employed, namely word masking. For each new vector that is being presented to the system only the words present in the given word vector are considered when computing the distance to the other words in the clusters. I.e. all words that are present in documents to be compared are discarded if they are not also present in the document to compare. This mechanism prevents unimportant, and most probably, unrelated words not to influence the distance measure. Without dimensionality reduction, or word masking, then the auxiliary words may unnecessarily add to the distance between two word vectors that in practice are quite similar.

The word vectors are clustered using the K-means algorithm . a classic and widely known and effective clustering algorithm (see [5]. In clustering the words are represented as vectors in the word space, and the purpose of the clustering algorithm is to assign word vectors that are similar to the same cluster in the vector space and assign word vectors that are different to different clusters. Two vectors are similar if the distance between the two vectors is small, and two vectors are dissimilar if their distance is large. One popular distance measure is the Eucludean distance. The K-means algorithm works as follows: a set of vectors is to be clustered into K clusters. Initially, the vectors are assigned arbitrarily

to the K clusters. Then the mean vector for each cluster is computed. Then, the vectors are reassigned to the cluster to which they are closest and the cluster means are recomputed. The process is repeated until some convergence criteria are met.

Finally, the results of the clustering algorithm are presented to the user as a pre-catalogued set of messages. Each time a new message or a group of messages arrives into the system the process is repeated incorporating the new messages into the process.

4 Quality measurements

It is relatively easy to assess the quality of classification tasks when they are applied to a training set, as this is a form of supervised learning, where the categories or clusters are predetermined. One can simply compute the success rate as the number of messages that are correctly assigned a cluster, and the error rate as the number of messages that are incorrectly assigned. However, assessing the quality of clustering is more difficult as there is no given mapping between the training category and the assigned cluster. We therefore deployed the following strategy: Each document di belongs to a training category ci and after the algorithm is deployed it is assigned a cluster ki. After training a category-to-cluster matrix is established where the columns represent the training categories and the rows represent the assigned clusters. An element at column i and row j denotes the number of documents from category ci that has been assigned cluster kj.

The quality measures are then computed in three steps. First, for each category the entire column is scanned for the largest value and this is marked as a category-to-cluster mapping. Second, for each cluster the entire row is scanned for the largest value which again is marked as a category-to-cluster mapping. If other elements in the row are also marked as a category-to-cluster mapping (in the first step) then these are remarked as "undecided". Third, the three quantities are computed as follows: the success rate is computed by summing all elements marked as category-to-clusters and dividing by the total number of documents, the failure rate is the sum of all non-zero unmarked elements divided by the total number of documents and finally the ratio of ambiguous messages is the sum of all elements marked as "undecided" divided by the total number of documents.

5 Test suites

In order to assess and document the effectiveness of the system through a repeatable experiment one is dependent on a test suite, with pre-categorized messages. The Reuters-21578 Text Categorization Collection [19] is a well-known and widely cited test suite, comprising of Reuters news articles from 1987 that has been classified and indexed by experts and later made available for research purposes. These news articles comprise medium to long well written pieces of English text. A news report is long compared to e-mail messages that often are short and poorly written with spelling mistakes and various abbreviations. The Reuters collection is therefore not completely representative of the problem domain. Further, we were unable to use this test suite as the current implementation of DIGIMIMIR is optimized for Norwegian. To manually categorize messages is time-consuming, difficult and error-prone. We therefore deployed two strategies for obtaining test-suites:

First, a small hand crafted test suite, comprising of 100 messages, was used for early testing. This set contains manually categorized fictitious messages, characterized as being easy to cluster and classify.

Second, the students at Oslo University College were used to create the second set of messages, comprising 160 messages. Four themes with 10 entries each were created. Each entry comprised a picture and a statement, such as a picture of well-known politicians or some hi-tech device. The students were asked to submit a question related to the entry via a web form. Thus the questions received could therefore be tagged with the given category.

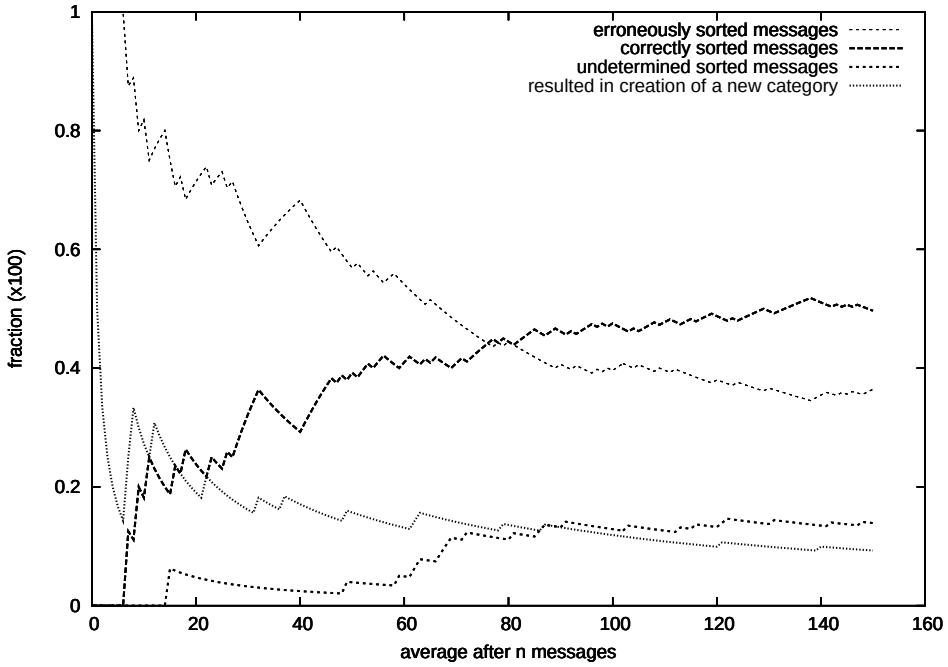

Figure 1: Accumulative success rates, error rates, uncertainty rates and clustering probabilities using the quiz test suite in unsupervised mode.

Figures 1, 2, 3 and 4 show the results obtained running DIGIMIMIR on two datasets. The figures illustrate the accumulative performance of the system, i.e. how the system state changes as messages are added to the system. The horizontal axes indicate the number of messages in the system and the vertical axes represent percentage correct answers, incorrect answers, ambiguous or uncertain answer and new clusters. Figure 1 shows the results obtained using the quiz dataset in unsupervised mode (160 messages), i.e. without training, and Figure 2 shows the results using the quiz dataset in supervised mode (80 messages), i.e. with training. One half of the dataset was used for training and the other half was used during testing. Figure 3 shows the hand-crafted dataset in unsupervised mode and Figure 4 shows the hand-crafted dataset in supervised mode. The messages were shuffled into pseudo random order for all the test runs.

All the figures show similar trends, namely that the success rate, the error rate and the percentage of new clusters converge as more messages are added to the system . and this is adhering to expectations. The quiz data reveals a clear difference between the unsupervised and the supervised runs. In unsupervised mode we achieve a successful classification rate of nearly 50% and an error rate of just below 40%. The percentage of ambiguous or uncertain messages is converging quickly to approximately 15%. Further, the probability percentage of generating a new cluster converges early at just over 10%. The results achieved in supervised mode are nearly twice as good. First, the success rate converges at around 80%, which is nearly a doubling in quality compared to unsupervised

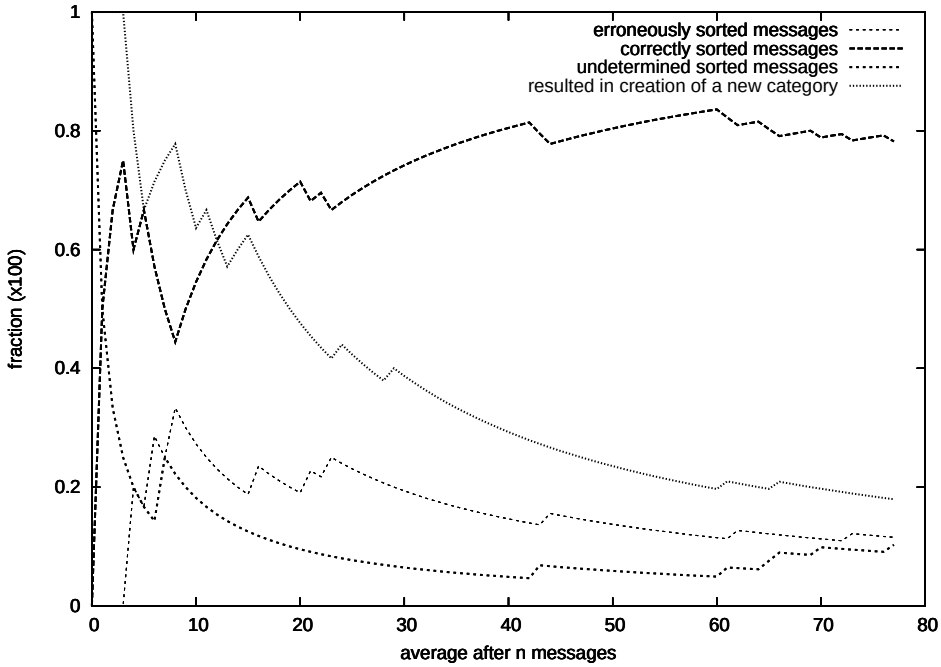

Figure 2: Accumulative success rates, error rates, uncertainty rates and clustering probabilities using the quiz test suite in supervised mode.

mode. This result is consistent with similar experiments reported in the literature on document classification. Second, the failure-rate is converging at 20%, which again is a halving in the number of incorrectly classified messages compared to unsupervised mode. Further, the rate of ambiguous or uncertain messages converges early at around 10%. It is also interesting to observe that the probability of generating a new cluster is converging much slower in supervised mode compared to unsupervised mode and that it is converging at a higher value of approximately 20% compared to 10% for unsupervised mode.

The shape of the graphs in the figures smoothly either decrease (error-rate and clustering probability) or increase (success-rate), but at certain points there appear to be discontinuities in the graphs resulting in temporal setbacks. These discontinuities mark the arrival of very dissimilar messages that result in new clusters being established. When new clusters are added the classification landscape is altered and leads to a temporary classification instability and slightly lower success rates. These messages can be genuine and naturally belonging in new clusters or they may simply be irrelevant or noisy messages.

Similar observations can be made in Figures 3 and 4. Figure 3 shows the accumulative success, error and clustering rates for the custom-made test data in unsupervised mode with very high success rates, and Figure 4 shows the same dataset in supervised mode.

6 Implications of automated e-mail support

The deployment of automatic document classification technology in the context of sorting incoming e-mails and automatically providing answers must be done with care. Nearly all requests are unique and the quality of the responses is best maintained by handling the requests manually. However, the quality of a support service is a tradeoff between the quality of the response content and its timeliness. A support department that does

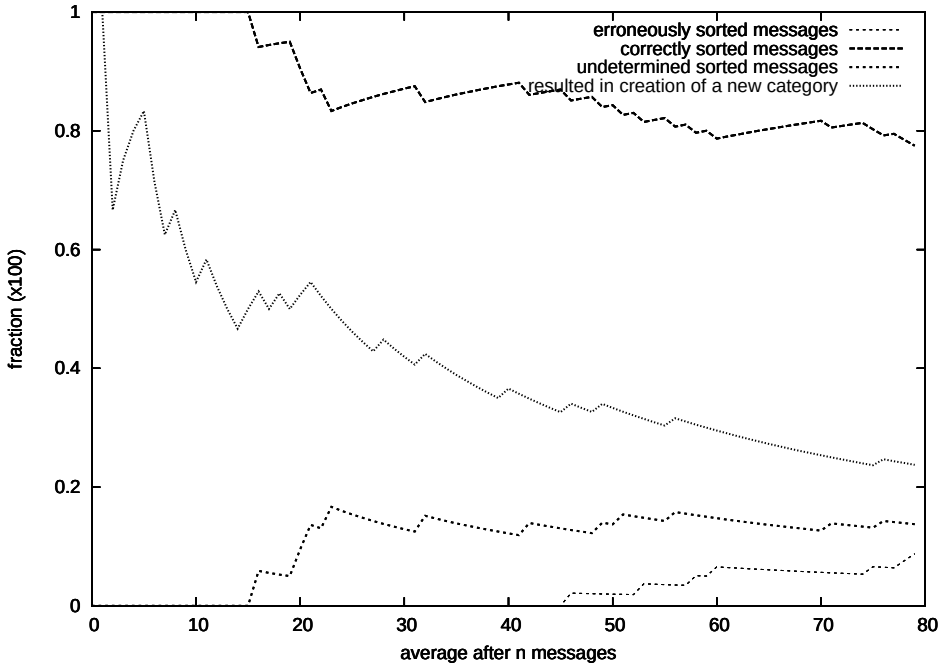

Figure 3: Accumulative success rates, error rates, uncertainty rates and clustering probabilities using the hand crafted test suite in unsupervised mode.

not respond to requests or the responses are answered weeks after they were originally sent can be frustrating to the users as they do not feel that their request is taken seriously. On the other hand, a rapid meaningless or obviously auto-generated incorrect reply can be equally frustrating and infuriating to the user and embarrassing to the organization. This may result in aggressive users. It is very difficult to make a foolproof system, i.e. a system that never incorrectly classifies a message and respond with the answer to a different question. To minimize the damage one can adopt psychological techniques, such as using humble wording in the responses. For instance, in DIGIMIMIR we used the following careful wording in the auto-generated replies: .We found that your question X is very similar to question Y. One answer to question Y is Z. Note that this response is automatically generated and a human will evaluate this response hopefully quite soon..

The optimal policy is probably a mixture of manual responses and automated responses. Manual responses should be used during low-peak periods for messages with a lower classification probability, while the automatic response mechanisms are to be used during peak hours when there is not sufficient manual resources to handle the stream of incoming messages. The administrator can then later inspect the requests that arrived during the peak time and assess the responses, and then send corrections to users when appropriate. Such a post-peak period inspection is still more efficient than having to respond to every message manually. It will in some situations be easier to spot a message that is out-of-place when it is placed together with other messages that are related. Ultimately, the inspection cycle is necessary as the 10 to 20% messages are incorrectly classified and needs to be handled manually. If one receives 500 messages a day, then this will account to 50 messages, and if the organization receives 5000 messages a day, this will obviously account to as much as 500 messages.

Politically speaking, an automated system is desirable as it is resource saving. As long as the financial gain of deploying such a system are larger than the negative impacts of the

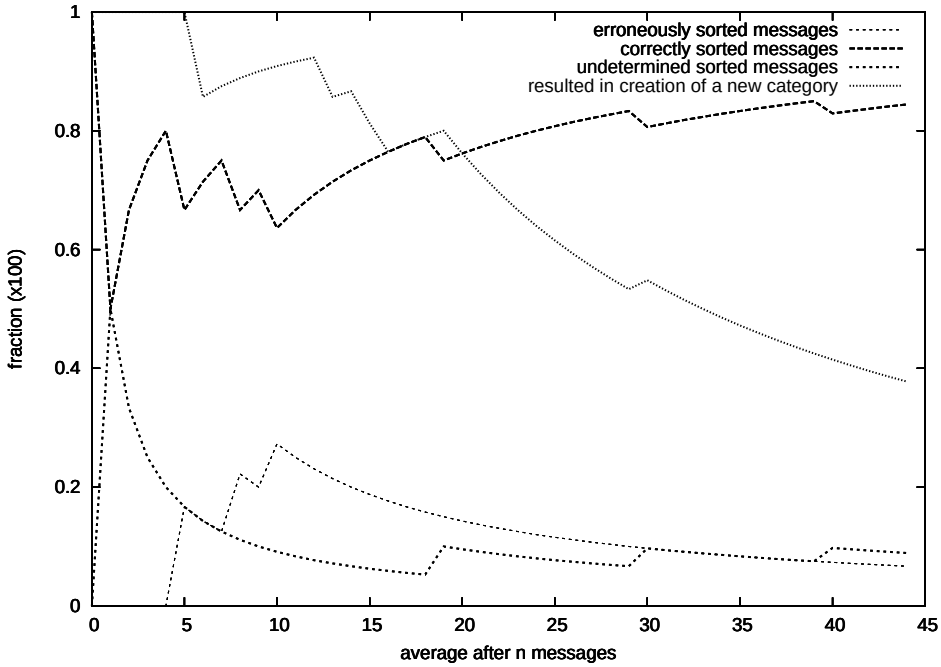

Figure 4: Accumulative success rates, error rates, uncertainty rates and clustering probabilities using the hand crafted quiz test suite in supervised mode.

errors introduced an organization is likely to embrace such technology. A consequence of this is that in the next instance the system administrators may be budgeted with even fewer resources by the decision makers.

7 Conclusions

In this paper we have addressed the problem of e-mail helpdesk support. Text-mining techniques were explored as means of partially automating the support tasks and a tool dedicated to this task was presented. Our experiments confirm that it is possible to achieve 50% or more correctly classified messages in unsupervised mode and 80% correctly classified messages in supervised mode. This can greatly help support staff reduce their workload, especially when combined with an auto-response feature. In operation, the system can exploit a mixture of supervised and unsupervised mode. Messages that are manually approved or classified messages can be used in a supervised manner, while new clusters can be established dynamically and unsupervised in order to get clear overview reports of totally new situations. We believe that document classification technology to a greater extend will be incorporated into the broad range of e-mail handling systems in the years to come due to the great potential for reducing the workload, but to ensure quality there should be a human in the loop. Further, such technology could also help reduce the emergency response time of a support time as emergency messages can be more quickly identified and separated from less urgent requests.

References

[1] J. Koutias Androutsopoulos, K. V. Chandrinos, G. Paliouras, and C. D. Spyropoulos. An evaluation of naive bayesian anti-spam filtering. *In Proc. of the workshop on*

Machine Learning in the New Information Age, 2000.

[2] S. Chakrabarti. *Mining the Web: Analysis of Hypertext and Semi Structured Data*. Morgan Kaufmann, 2002.

[3] K. H. Chang, R. Raman, W. H. Carlisle, and J. H. Cross. A self-improving helpdesk service system using case-based reasoning techniques. *Computers in Industry 30 (2)*, pages 113–125, 1996.

[4] M. Crispin. Internet message access protocol - version 4rev1. http://www.imap.org/, 2003.

[5] M. H. Dunham. *DATA MINING: Introductory and Advanced Topics*. Prentice Hall, 2002.

[6] R. Elling and M. Long. User-setup: a system for custom configuration of user environments, or helping users help themselves. *Proceedings of the Sixth Systems Administration Conference (LISA VI) (USENIX Association: Berkeley, CA)*, page 215, 1992.

[7] S. Foo, S. C. Hui, and P. C. Leong. Web-based intelligent helpdesk-support environment. *International Journal of Systems Science 33 (6)*, pages 389–402, 2002.

[8] S. Garfield and S. Wermter. Recurrent neural learning for helpdesk call routing. *Lecture Notes in Computer Science 2415*, pages 296–302, 2002.

[9] Google.com. Welcome to gmail. http://gmail.google.com/, 2004.

[10] C. Govindarajulu. The status of helpdesk support. *Communications of the ACM 45(1)*, pages 97–100, 2002.

[11] J. Graham-Cumming. Popfile trust the octopus. http://popfile.sourceforge.net/.

[12] T. Guy. Backstage at the helpdesk. *Proceedings of the 1999 ACM User Servive Conference*, pages 225–227, 1999.

[13] T. K. Ho. Fast identification of stop words for font learning and keyword spotting. *Proceedings of the 5th Int'l Conference on Document Analysis and Recognition*, pages 333–336, 1999.

[14] J. Justeson and S. Katz. Technical terminology: Some linguistic properties and an algorithm for identification in text. *in Natural Language Engineering 1:1*, pages 9–27, 1995.

[15] A. Kilgarriff. Putting frequencies in the dictionary. *International Journal of Lexicography 10:2*, pages 135–155, 1997.

[16] K. Kukich. Techniques for automatically correcting words in text. *ACM Computing Surveys 24:4*, pages 377–439, 1992.

[17] V. A. Kulyukin, K. J. Hammond, and R. D. Burke. An interactive and collaborative approach to answering questions for an organization. *Technical report TR-97-14", Dept. Computer Science, University of Chicago*, 1997.

[18] M. Lenz and H. D. Burkhard. Cbr for document retrieval - the fallq project. *Case-Based Reasoning Research and Development (ICCBR-97), Lecture Notes in Artificial Intelligence No. 1266, Springer-Verlag, Berlin*, pages 84–93, 1997.

[19] D. D. Lewis. Reuters-21578 text categorization test collection distribution 1.0. http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html, 1997.

[20] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using em. *Machine Learning 39 (2-3) May*, pages 103–134, 2000.

[21] L. Phillips. Hanging on the metaphone. *Computer Language, 7:12*, pages 39–43, 1990.

[22] M. F. Porter. An algorithm for suffix stripping. *Program 14:3*, pages 130–137, 1980.

[23] L. Salvadori. Gnats revisited. *Sys Admin: The Journal for UNIX Systems Administrators 6(8)*, pages 53–55, 1997.

[24] R. Shalhoup. A web-based helpdesk. *Sys Admin: The Journal for UNIX Systems Administrators 6(9)*, pages 41, 42, 44–46, 1997.

[25] E. Sneiders. Automated questioning answering: Template based approach. *PhD Thesis, Dept. Computer and System Science*, 2002.

[26] J. W. Wilbur and K. Sirotkin. The automatic identification of stop words. *Journal of the American Society for Information Science 18*, pages 45–55, 1992.

[27] W. Winwarter. Collaborative hypermedia education with the viena classroom system. *In Proc. 1st Australasian Conf. on Computer Science Education ACSE'96*, pages 337–343, 1996.

[28] W. Winwarter, O. Kagawa, and Y. Kambayashi. Applying language engineering techniques to the question support facilities in viena classroom. *Database and Expert Systems Applications*, pages 613–622, 1996.

[29] W. S. Yerazunis. The spam-filtering accuracy plateau at 99.9% accuracy and how to get past it, 2004.

[30] M. Zhao, C. Leckie, and C. Rowles. An interactive fault diagnosis expert system for a helpdesk application. *Expert Systems 13 (3)*, pages 203–217, 1996.

[31] N. Zhong, T. Matunaga, and C. N. Liu. A text mining agents based architecture for personal e-mail filtering and management. *Intelligent Data Engineering and Automated Learning - IDEAL 2002, Lectures Notes in Computer Science 2412*, pages 329–336, 2002.