UDC 004.75

A MACHINE LEARNING APPROACH TO SERVER-SIDE ANTI-SPAM E-MAIL FILTERING [1] [2]

I. Mashechkin, M. Petrovskiy, A. Rozinkin, S. Gerasimov

Computer Science Department, Lomonosov Moscow State University,
Building 2, MSU, Vorobjovy Gory, 119992, Moscow, Russia
Phone: +7-495-939-17-89, fax: +7-495-939-19-88
Email: mash@cs.msu.su, michael@cs.msu.su, andrey@mlab.cs.msu.su, gerasimov@mlab.cs.msu.su

Spam-detection systems based on traditional methods have several obvious disadvantages like low detection rate, necessity of regular knowledge bases' updates, impersonal filtering rules. New intelligent methods for spam detection, which use statistical and machine learning algorithms, solve these problems successfully. But these methods are not widespread in spam filtering for enterprise-level mail servers, because of their high resources consumption and insufficient accuracy regarding false-positive errors. The developed solution offers precise and fast algorithm. Its classification quality is better than the quality of Naïve-Bayes method that is the most widespread machine learning method now. The problem of time efficiency that is typical for all learning based methods for spam filtering is solved using multi-agent architecture. It allows easy system scaling and building unified corporate spam detection system based on heterogeneous enterprise mail systems. Pilot program implementation and its experimental evaluation for standard data sets and for real mail flows have demonstrated that our approach outperforms existing learning and traditional spam filtering methods. That allows considering it as a promising platform for constructing enterprise spam filtering systems.

Introduction

It is well known that from 40% up to 80% of all electronic messages in the Internet is spam. Spam, by definition, is unsolicited bulk e-mails. In other words spam is electronic messages posted blindly to thousands of recipients. Obviously, unauthorized e-mails mean real expenses for companies and personal users.

Nowadays various spam-preventing techniques have been developed. They can be divided into two major categories. The first one includes administrative and technical methods, which try to stop spam distribution. They are laws, which restrict sending of spam messages, new protocols for e-mail services based on authorized confirmation of the mail transfer like Sender ID standard [7], payments for each sent message, blocking of mail servers, which are used to send spam and so on.

The other category includes methods that prevent spam receiving by users, so called spam filtering. These methods can be also divided into two groups: traditional, which use fixed set of rules or signatures for spam filtering; and adaptive, which are based on statistical methods and artificial intelligence. Many traditional methods use different types of black lists of spam senders' addresses [8]. Traditional methods also use knowledge bases of keywords, rules and signatures of spam messages. These knowledge bases are prepared manually by experts and require regular updates. The systems based on such methods usually have low spam detection rate (60-70%). Besides, it is necessary to upgrade knowledge bases regularly to keep them up-to-date. So, these systems depend on efficiency of the updates' provider and they are unprotected during the period between new spam appearing and knowledge base updating. Moreover, traditional methods are not personalized, so they do not take into account peculiarities of the correspondence of a particular user. All this also decrease the accuracy. Nevertheless, the systems using black lists are widespread because of simplicity of their installation and maintenance. However, recently they are strongly criticized for the high false-positive error rate. The reason is that some providers use very simple and inconsequent rules to update and maintain their black lists.

Intelligent methods are a relatively new trend in spam detection. They may eliminate disadvantages of the traditional methods. Intelligent methods use statistical and machine learning algorithms. The algorithms are capable to classify mail into several categories using a statistical or machine learning models constructed beforehand on the basis of the precedent information [13].

To make such system work properly, it is necessary to train it on a set of e-mails that have been already classified as spam or legal messages. This training's result is a model that is then used for a new mail classification. Nowadays the most popular intelligent method for spam detection is Naïve-Bayes method [9]. Naïve-Bayes method is being implemented and is successfully used in several spam-detection systems [2, 4].

Intelligent methods have several advantages in comparison with the traditional ones. They do not depend on external knowledge databases and do not need regular updates. They do not use specific features of particular language, so they are multilingual. They are able to adjust the models using new samples of spam without the administrator's assistance and they can build personal filtering models.

[1] The research is supported by RFBR grant # 05-01-00744

[2] The research has been accomplished in cooperation with Advanced Algorithms, Inc. http://www.advalg.com

Nevertheless, despite their efficiency and intelligence these methods are not widely used in spam-detection systems at the enterprise level for several reasons. First of all, most intelligent methods are not stable enough when detecting legal mails and have a rather high level of false-positive errors. Intelligent methods have higher hardware requirements because they are based on computationally expensive algorithms.

The aim of our research is to offer a comprehensive e-mail-classifying solution for enterprise-level system that will be based on the intelligent analysis of messages. The solution should have the advantages of intelligent methods such as personification and high spam detection rate at low quantity of false-positive errors. At the same time the system should provide necessary efficiency to be used on enterprise-level mail servers.

Solution

Our solution is based on the intelligent classification algorithm that allows reaching necessary quality on the one hand, and on a multi-agent architecture that provides necessary efficiency, on the other.

For solving the classification problem we are using a statistical method based on support vector machines (SVM) [10, 11]. This method was applied to text categorization task earlier [5]. It is necessary to solve two problems to apply SVM for spam detection task: select proper kernel-function and find appropriate representation of e-mails as feature vectors.

We have selected the following representation for electronic messages: a feature set is defined as a set of all words that appeared in all analyzed messages more than the predetermined number of times. Furthermore, feature set is reduced by eliminating a set of predefined stop-words. Additionally, the feature set is expanded with features defined for all file extensions of files attached to the analyzed messages [12].

So, each message is represented as a subset of feature set. Each element of the set is a number of appearances of a particular feature in a message normalized by quantity of message's features.

We have carried out several experiments with various standard kernel-functions and have discovered that RBF kernel-function shows quite good results. It provides a high level of accuracy and comprehensible efficiency of the algorithm.

Besides, the solution should meet the following basic requirements: high efficiency; enterprise level; the ability to take into account personal features of each user's correspondence; platform independence; scalability; safety and privacy. These requirements lead us to a multi-agent architecture for the system. The general architecture of the system is shown on the figure 1.

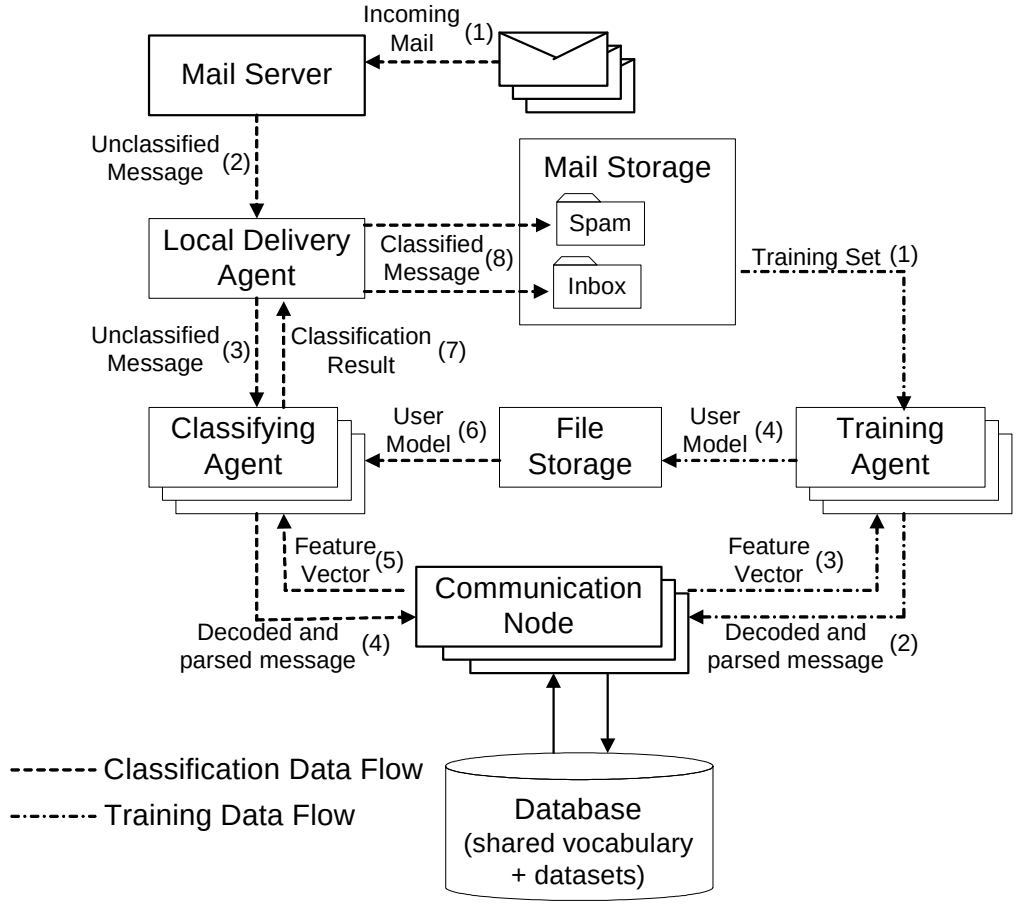

Fig. 1. Architecture of Enterprise Spam Detection System

The central communication node of the system is presented by one or several web-servers. It provides communication environment for training and classifying agents, supports shared vocabulary, converts messages to feature sets and provides GUI for users. The communication node stores shared vocabulary, temporary feature vectors and some additional user's information in the database. All time-consumptive operations like preprocessing and downloading messages, training user models and classification are moved to corresponding agents.

The training agent is a process that analyses user's messages and builds user's personal model on the basis of this analysis. The training agent allows customization for different message storages. In current version it is located at the centralized mail server and accesses personal data using IMAP protocol. Another solution might be the personal agent on a user's workstation that uses local mail storage from the personal folders. The common training workflow is the following. A user initializes training procedure using web-based interface. The central communication node starts training agent and initializes downloading the subset of user's messages using IMAP protocol. The training agent decodes each message, parses it into terms and then passes vector representation of each message to the central communication node using https protocol. The central node creates the feature vector for the message, saves it to the database temporarily and updates shared vocabulary located in the database. Have all messages been downloaded, the central node creates a training set from feature vectors stored in the database, saves it to the server's file system and starts training procedure using the created training set. The result of training is the personal user's model that is saved to the file system.

The classifying agent intercepts messages from the mail server and classifies them. The common classifying workflow is the following. When a new message arrives, it falls to the local mail delivery agent. This agent transfers the body of the message to the classifying agent. Then it decodes and parses the message and passes it's vector representation to the central communication node. The central node forms the feature vector from the message on the basis of the shared vocabulary and returns the feature vector back to the classifying agent. The classifying agent evaluates message's score using saved user's model and created feature vector of the message. The resulting score retuned to the local mail delivery agent that adds a status (spam / not spam) to the header of the message and then moves it to the appropriate user's mail folder. So, if a user reads mail using IMAP protocol he or she sees only legal messages in Inbox folder. Spam messages are not deleted, and are saved in a separate folder, that is also accessible for a user through IMAP. By default, IMAP Inbox and Spam folders are used as sources of legitimate and spam training sets for training user models. Once the system has been trained, a user may clear his/her spam storage to save disk space.

Adding new mail servers or new mail clients can extend the system functionality. To support them it is necessary to implement a new training or classifying agents. It allows creating uniform and well-scaled corporate system of spam filtration that unites heterogeneous company's infrastructure including different mail clients and mail servers.

The presented architecture was tested on the enterprise spam-detection system. The current version of the system has been tested with the following configuration: RedHat Linux 9.0 operating system, mail server Sendmail 8.2 or Exim 4.34, local mail delivery agent Procmail 3.22. Current versions of classification and training agents are written as C++ executables. We used mySQL database engine for the pilot system. The current version of the central communication node is implemented on PHP and based on Apache web-server.

We propose a solution that solves one of the main problems of learning algorithms – the resources consumption. This goal was achieved due to the multi-agent architecture that allows scaling the system according to real needs and unites heterogeneous infrastructure of an enterprise with different mail servers and clients.

Experiments

The purpose of our experiments is to compare efficiency of the proposed algorithm to other up-to-date methods and algorithms. The framework of the experiment includes two different tests that estimate classification accuracy.

The first test was carried out with the commercial product Kaspersky Anti-Spam Enterprise Edition [6]. This product uses traditional filtering methods based on the heuristic analysis of the text and headers of messages, and regularly updates knowledge bases. The experiment was carried out in the real-world situation, using the real dataflow from existing mail accounts. The experiment should show the superiority of the system based on intelligent methods of mail analysis, and it also estimates the performance and reliability of our solution in the real-world conditions.

The second test was carried out with the implementation of Naïve-Bayes method on standard test corpuses of messages. We used implementation of Naïve-Bayes method from freeware anti-spam solution SpamAssassin [2]. SpamAssassin has been customized so that Naïve-Bayes method took part in classification only. The experiment was carried out with several public test corpuses of messages. The purpose of the given experiment was to compare our algorithm with the most popular learning method.

Experiment #1: Kaspersky Anti-Spam Enterprise Edition. Kaspersky Anti-Spam Enterprise Edition [6] is a spam-filtering system for a mail server. It is based on several levels of spam identification such as linguistic analysis, spam signatures, RBL-services and so on. The support team prepares updates of the knowledge base every two hours.

The experiment was carried out on a real mail dataflow that had been gathered from several mail accounts. The flow of messages was copied to several mail accounts, which were processed by anti-spam filters. In total, 2700 messages were processed (2598 – spam /102 – legal mail). Four mail accounts were created, one - for Kaspersky Anti-Spam filter, and the three others - for the anti-spam filter based on our algorithm.

All parameters of Kaspersky Anti-Spam filter, except RBL-lists, remained as default. We considered that it was more correctly to switch off RBL-lists. One of the goals of this test was to discover the influence of size and structure of initial training set on the classification quality. Therefore, three different accounts were created for the experimental anti-spam filter. The initial training sets were:

Account #1: 200 legal / 200 spam

Account #2: 200 legal / 2000 spam

Account #3: 2000 legal / 2000 spam

Messages for initial training sets were randomly collected from previously received messages for those accounts. The additional training using newly arrived mail was not performed.

Results for Kaspersky Anti-Spam Experiment Table 1

Account	Initial training corpuses (spam / legal)	Detection Rate optimization		False Positive optimization	
		Detection Rate %	False Positive Rate %	Detection Rate %	False Positive Rate %
Experimental #1	200 / 200	100	7,8	88,6	0
Experimental #2	200 / 2000	100	9,8	99,9	0
Experimental #3	2000 / 2000	100	0	100	0
Kaspersky Anti-Spam	-	72,5	0	72,5	0

After finishing the experiment we processed the results and selected two different optimal thresholds for each mail account. One of them is for a minimized number of the false-positive errors and the other is for a maximized accuracy of detection. The results are presented in the Table 1. The results for Kaspersky Anti-Spam Enterprise Edition were more or less typical for today's traditional anti-spam filters. Its detection level is about 70%. The analysis of the results shows that it is possible to set parameters of our algorithm to achieve zero quantity of false-positive errors. Thus, the corresponding detection rate will depend on the size and quality of initial training set. Our anti-spam filter achieved better results than traditional methods system even with a smaller initial training set. The absolute accuracy was reached on a mail account with the maximal initial training set. There were no errors at all during two weeks of testing.

Experiment #2: Naïve-Bayes method. Two public mail corpuses were used for comparing our algorithm with Naïve-Bayes method from SpamAssassin anti-spam filter.

LingSpam Corpus [1]

Initially there are four versions of LingSpam corpus. We used 'bare' as the most general version of the set. A corpus's message contains body and subject only, but there is no header. The size of the set is 2893 messages.

The set was randomly divided into 10 equal parts, each of which contained about 290 messages (240 normal and 50 spam). We held ten different iteration of the test and then combined the results. Nine parts of the test corpus were used for the training and one for testing during each iteration.

SpamAssassin Corpus [3].

Corpus messages are presented in full and all headers have been saved. The set consists of three parts: 'spam' (500 messages with spam); 'easy_ham' (2500 normal messages, which are easily detected as normal by anti-spam filters); 'hard_ham' (250 normal, but very similar to spam messages).

Four parts of the set were used for training and one part was used for testing. So, five iterations were made with this test corpus.

To evaluate the performance of the algorithms we used ROC (Receiver Operating Characteristic) curves. The results of several tests have been averaged and ROC-curves have been constructed for each test corpuses. Evidently, ROC-curve for our algorithm is above a curve for Naïve-Bayes algorithm for both LingSpam and SpamAssassin corpuses. That means that our algorithm completely outperforms Naïve-Bayes method from SpamAssassin.

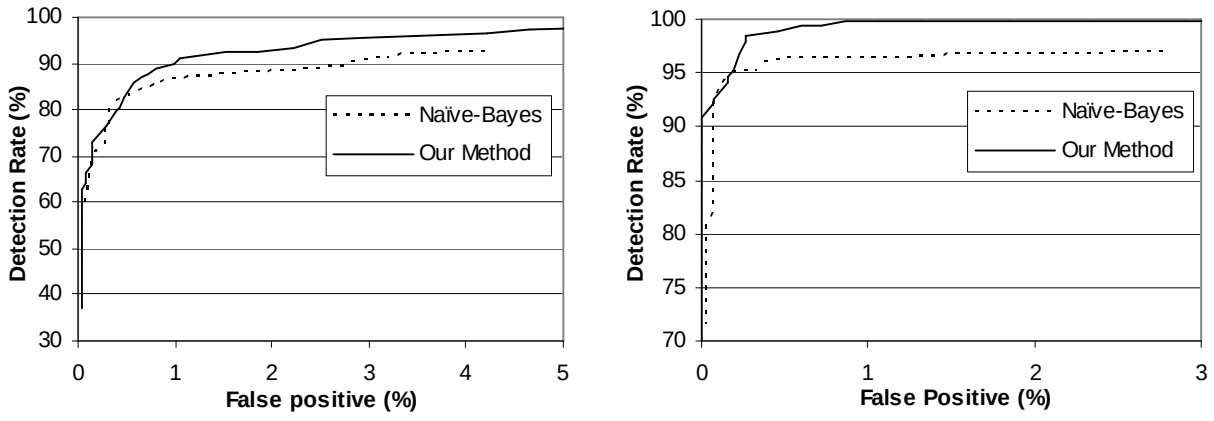

Fig. 2. ROC-curves for LingSpam Test Corpus

Conclusions

The developed solution offers precise and fast SVM based algorithm with better classification quality than Naïve-Bayes method's that is the most widespread now. It allows achieving high detection level. The problem of time efficiency that is typical for learning algorithms is solved by using multi-agent architecture that allows scaling system easily and building uniform corporate system for spam detection based on heterogeneous enterprise mail system.

Pilot program implementation and its experimental evaluation for standard data sets and for real mail flows have demonstrated that our approach outperforms existing learning and traditional spam filtering methods. That allows considering it as a promising platform for constructing enterprise spam filtering systems.

1. Androutsopoulos, I., Koutsias, J., Chandrinos, K.V., Paliouras, G., and Spyropoulos, C.D. (2000). An evaluation of naïve Bayesian anti-spam filtering. In Proceedings of the Workshop on Machine Learning in the New Information Age, 11-the European Conference on Machine Learning (ECML 2000), Barcelona, Spain, pp. 9–17 http://www.aueb.gr/users/ion/data/lingspam_public.tar.gz
2. Apache Software Foundation (2004) The Apache SpamAssassin Project http://spamassassin.apache.org
3. Apache Software Foundation (2004) The Apache SpamAssassin Public Corpus http://spamassassin.apache.org/publiccorpus/
4. Farmer, J. (2004) SpamPal - Mail Classification Program http://www.spampal.org
5. Joachims, T. (1998) Text Categorization with Support Vector Machines: Learning with Many Relevant Features. Proceedings of {ECML}-98, 10th European Conference on Machine Learning, Springer Verlag, Heidelberg, DE, 137-142
6. Kaspersky Labs (2004), Kaspersky Anti-Spam Enterprise Edition http://www.kaspersky.com/antispamenterprise
7. Microsoft Corp. (2004) Sender ID technology http://www.microsoft.com/senderid
8. ORDB.org (2004) Open Relay Database http://www.ordb.org
9. Sahami, M., Dumais, S., Heckerman, D., and Horvitz, E. (1998). A Bayesian approach to filtering junk email. AAAI Workshop on Learning for Text Categorization, Madison, Wisconsin. AAAI Technical Report WS-98-05
10. Scholkopf, B. and Smola, A., J. (2000) Learning with kernels: Support Vector Machines, Regularization, Optimization and Beyond. The MIT Press Cambridge, Massachusetss
11. Vapnik, V., N. (1998) Statistical learning theory, Wiley, New York
12. Yang, Y. and Pedersen, J., O. (1997) A comparative study on feature selection in text categorization. Proceedings of {ICML}-97, 14th International Conference on Machine Learning, 412-420
13. Yang, Y. (1999) An Evaluation of Statistical Approaches to Text Categorization. Information Retrieval, 1, 1/2, 69-90