A Community Based Approach for Spam Filtering

Deepak P,
Model Engg: College, Kochi, India
deepak-p@eth.net

Jyothi John,
Model Engg: College, Kochi, India
jyothijohn@mec.ac.in

Sandeep Parameswaran,
IBM Global Services India Pvt. Ltd., Bangalore, India
psandeep@in.ibm.com

Introduction

Current spam filters mostly base themselves on hard-coded rules like, "If a mail contains a red background or the word 'sex', it has such and such a probability of being spam". But different users have different ideas about what a spam mail is. For instance, a bald person may not consider a 'Prevent hair-loss mail' as spam while a journal editor might classify recurring 'submission-status' requests from the same author as spam. Further, anybody may regard constant requests from a stranger for help on his assignment as spam, even if it doesn't look like an 'unsolicited commercial bulk mail' and hence wouldn't be classified such by current filters. This paper describes an approach towards spam filtering that seeks to exploit the nature of spam messages that allow them to be classified into different communities. The new approach does not base itself on any prejudices about spam and can be used to block non-spam nuisance mails also. The approach inherently is user-centric, flexible and user-friendly.

The Approach

The Different Phases of the presented approach are summarized as below:

The phase of ignorance: Upon installation of the application, the system is ignorant of what spam is. The user has to mark the spam mails among the incoming ones and thus point to the system, 'hey, this is spam'. The system records the entire message. This continues until about 50 messages are accumulated.

The message similarity computation: One among the main algorithms to be used here is the computation of similarity between two messages. It uses heuristics such as 'add one to the similarity score if both have at least two common names in their "To" address'. Another method experimented was to represent a message as a vector of words occurring in it and taking the dot product of the vectors of the messages. Framing such mail similarity heuristics would require a lot of research into the general nature of spam mail.

The identification of communities: After accumulating close to 50 spam messages on the advice of the user, the system proceeds to identify communities of similar messages. It builds a graph with the messages as nodes and each undirected edge connecting two messages being labeled by the similarity weight between them. The system now finds strongly connected communities of mails based on some threshold.

Community Cohesion Scores and Signatures: It computes a score for each community which indicates the cohesion within the community. It is computed on the basis of some heuristics such as the sum of the weights of all edges within the community divided by the number of nodes in the community. The aim is to give high scores to communities of high cohesion. Community signatures consisting of a set of messages from the community would be identified. If a community consists of 3 sets of 10 identical messages each, the signature should consist of at least one representative from each set.

Spam Identification: Each incoming message is tested against the signatures of each spam community as to whether its inclusion would enhance the cohesion within the community. It is added to the community and marked as spam if it either increases the cohesion of the community or

0-7803-8482-2/04/$20.00 ©2004 IEEE.

has a very high similarity score with one or more of the community members. If not, it is marked as legitimate and passed to the user.

Maintenance: If the user opines that a message delivered to him as legitimate was actually spam (a false negative), it is added to the community to which it best fits or put as a single member community. Periodically, if there is a proliferation of small communities, those are gathered and processed just as the initial set of 50 odd spam messages to identify larger communities. If the user opines that a message marked as spam was legitimate (the dreaded false positive), the system inspects the communities to find messages of very high similarity with the one in question and they are deleted from the database. As more and more messages are identified as spam, they are added to the database. Periodical database 'cleaning' is done by considering communities, finding very dense subsets within them and deleting some of the messages which are connected to the communities by dense edges. This is extremely useful in purging identical messages from the set. Periodically, the system does a warm reboot, by dissolving all communities and identifying them from the entire set of messages afresh. A full reboot would obviously, be to empty the database.

Adaptation: Adaptability to changing nature of spam is to be taken care of. It would be done by the system by deleting communities that have had no admissions for a long time. Perhaps the user might have been taken off the spammer's list or the nature of spam sent by the spammer would have changed. In either case, holding the community in the database would be of no use. Further the user could be provided options to manually clean up or delete communities.

The system comes in with an empty memory and learns what spam is, from the user. The user is free to point to some nuisance mail (such as an old lover who is no more interesting) and mark it as spam. A person entertaining some special spam category, e.g., porn-spam, can continue to keep himself entertained by not marking them as spam during the ignorance phase. The system provides little help in the phase of ignorance, but it does not come in the way. In cases where spam comes to a user from only a few spammers, each community might get mapped to a single spammer. Further, as the system is implemented per user, the implicit rules may be more user-specific providing more flexibility.

The approach and the different heuristics used (such as those for message similarity computation) have been tested under spam mail collections. The heuristics and the approach as a whole were seen to have worked well. On the average, about 84.5% of total messages could be classified into communities and that indicates that this approach would definitely be feasible and would live up to the expectations.

It can be implemented as a mail client plugin, whereby the matching algorithms can be done at the client machine. Future work may be directed towards developing algorithms for spam message similarity computation, for selecting victims to be purged off to limit database size, to enable the system to self-adapt to the changing nature of spam mails, and algorithms for identification of communities from a corpus.