Self-Learning Spam Mail Filter

SM1-06

Group Member:
Lam Kin Yan (04041636)
Ma Chau Wing (04123159)
Wong Sau Nga (04110009)

Supervisor:
Professor Siu, Man Hung
**Introduction**

E-mail is one of the most convenient communication tools. People use it for many different purposes such as announcement, promotion, sharing information, etc.

However, there are spam mails which abuse the e-mail function. This is not only waste space in e-mail in-boxes, but also waste people’s time and money.

To solve the problem, spam mail filters must be used to filter out the spam mail.

**Aims & Objectives**

a) Filter features we focused on
   1) Self-learning - learn the spam features from all previous e-mail for further filtering.
   2) User adaptable - suit for different user.

b) Language we focused on - E-mails that have English content.

c) Target rate of false-reject and false-accept
   1) False-reject - less than 2%
   2) False-accept - less than 5%

**Design & Implementation**

To build the Self-learning filter, we first divided the e-mail into several parts - header, content, HTML, attachment and Yale. Then, we built separated filters for different parts.

Then, our filters will calculate and extract features into a features vector. The features will be passed to Yale for processing. Our program will then have a determination.


**Header Filter**

a) Headers to be checked - From, Sender, Return-Path, Received and Subject.

b) Things to be checked in headers - Validity of formats, breakage of routing path, Consistence of IPs, the White/Black list and Real IPs.

**Content Filter**

a) Tokenization - tokenize the contents by space and punctuation.

b) Unique word count per e-mail

**Spam Probability of each Word**

Both Content and Header filter used the Robinson’s technique together with biased version of Graham’s technique.

a) Biased version of Graham’s

\[
P = \frac{SH}{TS + 2 \times IH}
\]

b) Robinson’s Technique

\[
F(W) = \frac{SX + N \left( P(W) \right)}{S + N}
\]

**HTML and Attachment Filter**

a) Served as a complement of Content and Header filters

b) features looked on: opt out links, hyperlink that can identify user’s mailbox, hidden fonts(for Bayesian poisoning), embedded images and embedded HTML

**Scores Combination**

The filters use the following formulas to combine the scores.

a) Header filter - Inverse Chi-square

\[
P = 1 - ((1 - P1)(1 - P2)...(1 - PN))^\frac{1}{N}
\]

\[
Q = 1 - ((P1)(P2)...(PN))^\frac{1}{N}
\]

\[
s = \frac{1 + \frac{P}{P+Q}}{2}
\]

b) Content filter - Paul Graham

\[
P = \frac{ABC...N}{ABC...N + (1-A)(1-B)(1-C)...(1-N)}
\]

**YALE Filter**

This filter made use of the algorithm provided by the software called Yet Another Learning Environment.

The YALE filter classified an incoming e-mail by using the model. The model was a set of rules documenting how to classify an e-mail. After we had the rules, we could process (classify) e-mails as fast as 0.01 second per mail.
**Result**

Total e-mail used for Training and Testing
a) Training - 7000 Spam and 7000 Non-spam
b) Testing - 13000 Spam and 13000 Non-spam

**Content Filter**
1) Average false-reject rate is 4.7%
2) Average false-accept rate is 5.9%

**Header Filter**
1) Average false-reject rate is 4%
2) Average false-accept rate is 2.28%

**YALE Filter**
The result from testing algorithms in YALE demonstrate that we have made significant progress in classifying mails by extracting features from e-mails and identify them using previous data(model).
1) SVM - EER is 0.24%
2) Decision Table
   i. 0% false-reject
   ii. 0.00045% false-accept
3) Decision Stump - EER is 2.1615%

When the number of training set is small, we should use SVM, but when the training set is large, we should choose Decision Table instead.

**Conclusion**
From the test result, we found that, in general, the project has achieved two main goals - self-learning and spam filtering with correct rate more than 99%.