Cut Once - A Thunderbird extension for Recipient Prediction and Leak Detection

Vitor Carvalho(vitor@cs.cmu.edu)
William Cohen(wcohen@cs.cmu.edu)
Ramnath Balasubramanyan(rbalasub@cs.cmu.edu)

19th Jan, 2008

1 Description

Cut Once is an extension to Mozilla Thunderbird which is a popular open source email client. Cut Once implements methods from the following papers to perform recipient prediction and leak detection.

http://www.cs.cmu.edu/∼vitor/publications/papers/carvalho07sdm.pdf
http://www.cs.cmu.edu/∼itor/papers/ecir2008.pdf

The extension is entirely written in Javascript.

2 Usage

2.1 Installation

Thunderbird extensions are distributed as .xpi packages which can be easily installed by double clicking on the file icon.

2.2 Training

Cut Once has to be trained before it is able to make recipient predictions. Training is achieved by opening a "Sent" folder and hitting the "Train" button on the toolbar. The time taken for training depends on the number of messages in the folder, the speed of the processor etc but a rough estimate is 160 messages per minute. Once the train procedure is completed, a model file is created in the user's home directory which is then read in by Cut Once everytime Thunderbird starts up. A weekly reminder encourages users to retrain on a regular basis.

2.3 Model file

The model file created during the training process stores the following pieces of information about the user's Sent folder.

- **Centroids** A centroid for each email address to which a message was sent to is computed by calculating a mean vector over all the messages addressed to the email address. Each email is represented by a TFIDF vector over the words in the subject and body.
- **Document frequencies** A table of words and its corresponding document frequency which is the number of messages in which the word occurred. This is necessary to compute TFIDF vectors for messages during runtime.
- **Recency and Frequency Ranks** Candidate email addresses in the Sent folder are ranked by recency and frequency to establish a baseline ranking. The ranks assigned to each email address are saved in the model file to enable Cut Once to display a baseline ranking during runtime.

The training procedure trims the size of the model by discarding words whose document frequency is below a threshold and by discarding email addresses which have very few messages addressed to them.

2.4 Prediction

Cut Once recipient predictions can be seen in two different ways. In the first method, the user can explicitly seek recommendations by hitting the "Recipient Recommendations" button on the toolbar in the Compose window. Clicking a recommended email address adds the address to the recipient list in the Compose window.

In the second method, a dialog box pops up when the user hits the Send button. This dialog box higlights possible leaks (defined as email addresses that have been chosen as recipients by the user which are unlikely to be valid recipients for the message composed, based on the history of past communication with this address) and also lists other recommended recipients. A countdown timer ensures that the message is automatically sent after ten seconds if the user does not wish to use the dialog.

In both the methods, users are presented with two different ranked lists of recommended recipients. The ranked list of recommendations on the right provide a baseline ranking that is based on recency and frequency alone and is independent of the message content. The list on the right is ranked using a mean reciprocal rank score that combines ranking using recency, frequency and TFIDF(message content).

2.5 Logging

User actions within the recipient recommendations dialog and the dialog box opened after the Sent button is hit are logged. This includes information such

as the rank of a recommendation that the user clicks on, the time taken by the user to accept a recommendation, the position in the list of a leak that is removed by the user etc. However no personal information such as email content is logged. Users are asked every week if they would like to send this log file to the researchers who developed the extension. Log files can also be explicitly sent by hitting the "Mail statistics" button on the main window.

2.6 Code

The codebase for the extension involves the following types of files.

- **XUL files** overlay.xul, compose_overlay.xul, train_progress.xul, infoleakalert.xul and ccalert.xul. These specify the layout of UI windows.
- **JS files**:
  - overlay.js - contains the train function. The most important functions are trainPrevention() which controls the training, doMessage() which processes each email in the Sent folder and IMAPEnded() which computes centroids, determines recency and frequency ranks and dumps all necessary data into a model file. The execution path of the training code is a little hard to follow since we are dealing with an event driven model.
  - compose_overlay.js - contains functions that compute cosine similarity. The most important functions are recommend2() and predict().
  - infoleakalert.js - populates the dialog that pops up when the Send button is pressed.
  - ccalert.js - populates the Recommend Recipients dialog.

2.7 Comments

Please send all comments and suggestions to email.research.cmu@gmail.com