E-mail filing system design document

Alon Altman

July 12, 2001

1 Problem Description

The purpose of my project is to construct an automatic e-mail filtering system, based on the full text of the message, that will *not* require manual defenition of filtering criteria. Instead, the system will learn the filtering criteria based on previously (manually) filtered messages.

2 Use model

The system will consist of a perl script, that will be usable in three distinct ways: to initalize the system, as a filter, and to correct errornous filtering.

The system should be used after using rule-based tools (such as `procmail`) to sort all easily sortable messages before passing all remaining messages to the script. This could drastically improve performance, because you get 100% performance on easily sortable messages, and don't confuse the classifier with those folders.

2.1 Filter use

The script will usually be used as filter. In this use mode, the input of the script would be the message in standard e-mail message format and the action taken would be filtering the message into a folder as learned by the program.

Furthermore, under filter use the program will assume the filtering was correct and re-enforce its rules based on this assumption.

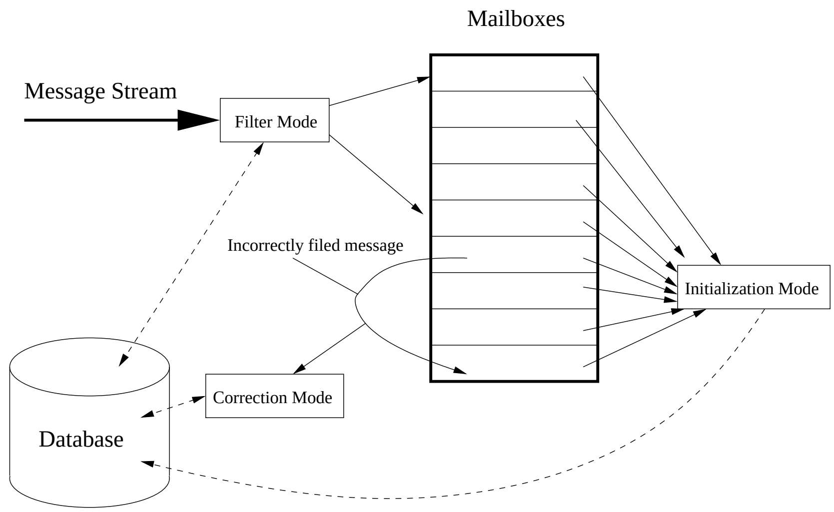
2.2 Correcting errors

If an error has occured in filtering, or a new folder is desired, the user will have to manually refile the message. During this refiling the user must also configure his or her mail program to start the script in correction mode. The mail program should also tell the script the original and new folders for the message.

The input in this mode is the message itself (passed via STDIN) and the original and new folders (passed via command line).

In correction mode, the script will undo the reenfocement done in during filtering and will update its database to reflect the corrected filtering.

Figure 1: Stucture of the system



2.3 Initalization use

In this mode, which should be used only once when the system is first used, the system will learn the user's existing mail folders, and define the filtering operations accordingly.

The input in this mode is a list of folders to be filed. The folders should be in standard mail folder format, and will be parsed message-by-message.

2.4 Feature selection

This mode, that should be used every period of time (e.g. daily), runs feature selection on the parsed data and creates a file with a feature list to be used.

Currently, it appears that feature selection does not improve the performance of the classification.

3 System Stucture

As seen in figure 1, the system operates in the three above-mentioned roles. During its operation it iterfaces with the database, which would be kept as a perl DBM file.

In order to operate the system the user should do the following:

1. If there are existing folders, run the script in initialization mode.

2. Setup that all new mail which cannot be filed by a rule-based filer such as `procmail`, be forwarded to the script in filtering mode.

3. Setup that the mail program pipe all messages that are moved between folders to the script in correction mode.

4 The database

The database will include the following data for each folder:

- Number of messages corresponding to each feature (See below)
- Total number of messages in folder.

This data can be easily used to calculate feature frequencies for each feature.

4.1 Word features

Words will be searched for in textual parts of the message (MIME type text/plain). Optionally, encodings such as quoted-printable and base64 will also be parsed. Another optional feature, is parsing of text/html, application/postscript and application/pdf attachments.

For plain text, the following characters will be categorized as text: [a-zA-Z0-9א-ת]. The following characters will be removed before parsing: [’‘’’_]. All other characters will be regarded as word delimiters.

URLs in plain text will be detected and marked as such, adding two word features - one valued as the method and host name (e.g. `http://www.somewhere.com`) and the second as the entire URL. This way URLs will be treated correctly.

4.2 Message features

Message features are determined from the message headers and attachments, and will be marked with capital 'H' for message features and capital 'P' for part features.

The message features I use are:

- Extended headers (x-*) are kept verbatim as features.
- To/Cc/From/Sender headers are kept both verbatim, and each e-mail address in the header seperately.
- Words from subject are extracted as above and kept as special features, but also as word features.
- MIME types of all pars are also kept as featues.