

# Using old Spam and Ham Samples to Train Email Filters

Jose-Marcio Martins da Cruz  
Centre de Calcul et Systemes d'Information  
Ecole des Mines de Paris  
Paris, France

Jose-Marcio.Martins@mines-paristech.fr

Gordon V. Cormack  
Cheriton School of Computer Science  
University of Waterloo  
Waterloo, Ontario, Canada  
gvcormac@uwaterloo.ca

## ABSTRACT

Email spam filters are commonly trained on a sample of recent spam and ham (non-spam) messages. We investigate the effect on filter performance of using samples of spam and ham messages sent months before those to be filtered. Our results show that filter performance deteriorates with the overall age of spam and ham samples, but at different rates. Spam and ham samples of different ages may be mixed to advantage, provided temporal cues are elided. The experiments on both corpora show that performance deteriorates faster when using only the body of messages than when using the whole message or headers only.

## 1. INTRODUCTION

Spam filters are commonly trained on historical collections of messages, each labeled as spam or ham (non-spam). Their theory of operation assumes that these training messages are a random sample of those to be filtered; an assumption that is clearly not true because, when the filter is trained, the set of messages to be filtered exists only in the future. It is known that future messages are best approximated by recent messages [4].

Under the assumption that models built with recent samples best represent what future objects will be, models should be continuously updated, including new samples and forgetting old ones. Over time, models may become inadequate for several reasons : class priors can change over time or underlining changes can happen on the nature of objects being studied.

However, acquiring and labeling recent messages may be impractical, and they may not be plentiful enough for adequate training. Sometimes, it may be more practical to acquire recent examples of one class than the other; for example, spam from a spam trap or ham from the client interface. Also, when filtering messages for a group of recipients, it may be difficult to continuously update the ham data set and it could be desirable to train the filter with a convenient initial data set of hams and spams and, over time, to preserve the initial ham data set and renew only the spam data

set.

In this paper we present some experiments we've done in order to understand what happens to the effectiveness of a filter being used in conditions which aren't ideal: less-than-recent training samples or samples having different ages. Also, we're more interested in understanding qualitative behaviours than absolute values of effectiveness, as the latter depends heavily on the kind of classifier being used and on the context.

This paper is organized as follows. Section 2 recalls some relevant research on how to update models used by classifiers to solve the concept drift problem in spam filters. Section 3 presents the objectives of our this paper. Section 4 presents the environment in which we've done our experiments. In section 5 we describe our experiments and present results. And finally sections 6 and 7 presents some discussion and conclusions.

Due to lack of space, experimental sections and results were omitted from this paper. A full version of this paper, including these sections, is available at :

<http://www.j-chkmail.org/ceas-09/gvcjm-ceas09-full.pdf>

## 2. RELATED WORK

It's usually accepted that email characteristics change over time [8]. While it can be assumed that people don't change frequently the way they write legitimate messages (hams), spam evolves for different reasons. The amount of spam changes to reflect spam activity[8], so class priors change accordingly. Spam filtering can be seen as an adversarial game [14] where the strategy of each part changes over the time : spam content change as spammers want to deceive spam filters (generating false negatives) and spam filters change to adapt to the changes in spam content.

It has been shown that spam filter evaluation is more accurate when using an on-line model, where messages are submitted in chronological order, than with a batch model where order is irrelevant [4]. This reflects the time dependency of email.

Changes in class (target) priors may be simpler to handle, as they may be observed directly. Changes appearing in hidden concepts [18], may be much harder to detect and may even be confounded with noise, as these changes can't be correlated to any directly measurable parameter. Changes in hidden concepts is usually referred as *concept drift* [18].

Machine learning techniques are applied to many domains handling non-stationary streams of data. Research in these domains is very active and probably even more than in the spam filtering domain. Data Mining research is intended to

identify and acquire information from streams of data [10] or even simply to detect when changes occur and the speed of change [1][3]. Clustering [2] and On-line Classification [19] of non-stationary data streams are domains nearer our spam filtering problem.

The canonical solution implies that all past relevant samples shall be available and some method exists to integrate new samples to the current model and to forget the oldest or less useful ones. The two hard points of this approach are the storage space needed to save past samples and the heuristic used to select which old samples can be removed, for which the computational cost may be high. This can be even harder when the amount of data being handled is large as well as the rate at which changes occur.

In the literature, techniques employed to handle concept drift (and to update models) are commonly studied in categories specific to the domains or the kind of classifiers to which they will be applied. We consider two categories, based on how they are applied to spam filters.

The first category, which we will refer as *instance-based update*, are those requiring that old samples, or at least a summary of each sample shall be available when the model will be updated. Forgetting old samples consists in just removing them from the training data set. A trivial example of this kind of solution is the use of a fixed size sliding time window ending just before the current date.

The second category, which we will refer as *incremental update*, are those for which each new sample is used to update the parameters of the model and is discarded just after. This is the kind of solution usually found in spam filters, even if not all classifiers are well suited for this.

Although, in theory, all new examples should be used to update the model, in practice it's common that only some are used. Most of the time it's impossible to have the correct label for each example, and sometimes it shall be considered that this feedback will be available for only one class. This happens in both situations described below.

## 2.1 Instance-based update

The first approach raises naturally from instance-based classifiers (sometimes referred to as "lazy learners"), where all examples may explicitly be used during the classification process [18] or to update the model when it was detected that the model changed. Within this approach, the learning process maintains all samples (or a summary of them) inside a time window of fixed or variable size. The learning process evolves moving the window forward, adding new messages to the front and removing old ones from the tail. Although this approach isn't limited to *instance-based classifiers*, we will refer to it as *instance-based training*, as it shall store and individually access each sample in order to remove older ones from the training set.

Using this approach, Cunningham [6] retrain a nearest neighbour classifier when the accuracy falls below some pre-defined level. Fernandes-Riverola et al [9] use feature selection to select which samples to remove or to maintain and to update the window size. Hsiao [13] detects changes inside clusters to decide when to update them. Delany et al [7] uses a case based classifier (lazy learner) : only misclassified messages are added to the data set and a periodic retraining is done to remove less relevant samples.

One advantage usually mentioned by defenders of *instance-based training* approach is the ability to detect and adapt to

local changes inside classes. On the other hand, the storage place needed to save all examples may be important. Also, if adding new samples may be trivial, unless using a simple fixed size time window, deciding which old samples can be removed may not be the same, particularly on large training sets.

## 2.2 Incremental update

In the opposite approach, incremental *update*, samples are presented sequentially for training and are discarded just after use. The goal is to eliminate the need of saving all past examples.

When doing active learning, the classifier is allowed to ask the label a sub set of the messages submitted to the classifier. Messages for which the real label will be asked may be chosen randomly or based on some criteria, e.g., messages for which the assigned score is close to the classification threshold [17][16].

Most open-source filters use some variant of "Train on Errors", "Train Until No Errors" or "Train on Everything". Bogofilter[15], an open source "Bayesian" spam filter, expires features (not examples containing these features) which weren't seen after some time. It's not clear that models updated with these approaches don't degenerate after some time, as it hasn't be shown that these approaches converge to the real models. Sculley[16] showed that some classifiers, updated with the "Train Until No Error" approach (repeat submitting the same misclassified sample until the classification is correct), present over-fitting and may be easily broken by noise.

Goodman and Yih [11] use sequential gradient descent in an adaptive logistic regression classifier to do sequential learning, eliminating some drawbacks of previous solutions.

## 2.3 Effects of concept drift

Although there had been many research work to find efficient ways to solve the concept drift problem, at our knowledge, very few research were done to evaluate the consequences of the drift itself when models aren't updated. Some limited results can be found in [7], but the methodology seems specific to the kind of classifiers being evaluated.

Examples of questions which remain without answer are : how does concept drift affects classification errors, at which class concept drift is more important or which characteristics of messages, other than the content itself, can mitigate these effects.

## 3. OBJECTIVES

Our main objective is to investigate the influence of concept drift in spam filter effectiveness. So, we're interested to identify the class for which concept drift is more important and if the two approaches to handle concept drift result in completely different effectiveness. A secondary objective is to understand at what extent a spam filter can be used, without being updated, to filter recent messages with an "acceptable" effectiveness.

The first two series of experiments try to simulate the two approaches, described in the previous section, used to update the email stream model. With these experiments we also try to identify some points which can help to mitigate the effect of concept drift in spam filtering. The last series of experiments was done to investigate at which part of email (headers or body), concept drift is more important. These

objectives are described below.

- *Temporal References* - temporal references are always present in email messages : most of them inside headers and sometimes in the body. Cormack and Lynam [5] suggested how some of them appear inside messages. In fact, the big picture here is the identification of features which will deviate the classifier from the originally intended targets : “*old and recent messages*” instead of “*hams and spams*”. If their effect isn’t negligible, it become necessary to identify these features and elide them to minimize their influence on the remaining experiments. This change in the classification target was discovered during our preliminary experiments and, given its importance, we decide to included them in the whole picture.
- *Instance-based update* - effectiveness of mail filtering is supposed to degenerate with the age of samples used to train the filter. With this series of experiments we examine the situation where, after an initial training, the filter is employed during some time without updating the training data. This situation is compared to one side training with replacement : only one class of training data sets is updated using a constant size time window (recent messages are added while old ones are removed) and is equivalent to a trivial *instance-based update*, described above.
- *Incremental update* - Investigate the effect of samples age when doing one-side or both sides incremental update. While in the previous objective old messages are replaced by new ones, we’ll just add new messages to the training set. We’re interested in the comparison of situations where both classes are updated, only one is updated and none of them are updated.
- *Whole message, headers and body* - The information present in the headers and in the body of messages aren’t of the same nature. Information inside headers are mostly related to the way the message was created and to its path from the sender to the recipient. The body of the message contains, most of the time, the real message payload, but may also include some meta-information, like attached files, or HTML code. Given this difference, it’s interesting to investigate if the influence of the age of samples is the same in both parts.

## 4. CONCLUSIONS

It has been usually assumed that characteristics of email, and mainly spam, change substantially over time and spam filters shall absolutely be continuously retrained. Most people don’t change the way they usually write legitimate messages so changes in ham mailboxes come most of the time from context changes : someone begins to work on a new project or subscribes to some unusual newsletters. Spam content changes continually and results from the “arms race” between spammers and and filter developers.

The use of old training data degrades performance, but not nearly so much as the use of raw training data in which the ham and spam have different ages.

Our experiments showed that if temporal references are removed, training data of mixed age may provide improved

performance in the situation where only new ham or new spam is available. Header removal is too radical as it dramatically compromises overall performance. If a few tell-tale temporal cues are identified and elided, substituting newer training data for one class of messages appears to yield improved effectiveness over using old for both. Our approach to identifying the training cues was not entirely automatic, and not entirely blind to the training data (but definitely blind to the test data). We believe it is a good candidate to be automated. And even if effected manually, it is much more efficient than labeling a new training set. The cues we discovered closely match those mentioned by the authors of the TREC 2005 Spam Corpus [5].

Experiments with MrX corpus had shown that, for this particular corpus, using old ham is worse than using old spam, while the same experiment with MrJ corpus doesn’t present a so noticeable difference. On the other hand, for both corpora, when both hams and spams are aged together, effectiveness doesn’t change too much.

We can’t generalize our results, as our experiments were done with only one kind of classifier (logistic regression), and only two corpora (MrX and MrJ), but it seems to us that, provided some preprocessing is done in training data such as removal of temporal references, it’s possible to use old samples or mix samples of different ages with limited effectiveness loss.

We also have shown that incremental update, on both classes ham and spam, may improve effectiveness but sigle-sided incremental update degrades the misclassification rate on the opposite class. However endless incremental update can eventually generate an overfitted model ([12] p. 194), mainly on classifiers unable to forget less recent samples.

These results aren’t completely surprising and, at some extent, they confirm assumptions which are usually accepted without being verified experimentally. On the other hand, the few experiments we’ve done with only two different corpora had shown that it may not be surprising to get some qualitatively different results in different contexts. (e.g. the degeneration with age we noticed when using only the body of messages).

These experiments also raised some questions which remain unanswered and deserves some more deeper research with other corpora and other classifiers.

To conclude, it seem to us that a better understanding of data being handled by spam classifiers, and the context, may improve the effectiveness of spam filters.

## References

- [1] AGGARWAL, C. C. On change diagnosis in evolving data streams. *IEEE Trans. on Knowl. and Data Eng.* 17, 5 (2005), 587–600.
- [2] AGGARWAL, C. C., WATSON, T. J., CTR, R., HAN, J., WANG, J., AND YU, P. S. A framework for clustering evolving data streams. In *In VLDB* (2003), pp. 81–92.
- [3] BOETTCHER, M., HOEPPNER, F., AND SPILIOPOULOU, M. On exploiting the power of time in data mining. *SIGKDD Explorations Newsletter* 10, 2 (2008), 3–11.
- [4] CORMACK, G. V., AND BRATKO, A. Batch and on-line spam filter evaluation. In *CEAS 2006: The Third Conference on Email and Anti-Spam* (2006).
- [5] CORMACK, G. V., AND LYNAM, T. R. Spam corpus

- creation for TREC. In *CEAS 2005: The Second Conference on E-mail and Anti-spam* (2005).
- [6] CUNNINGHAM, P., NOWLAN, N., DELANY, S. J., AND HAAHR, M. A case-based approach to spam filtering that can track concept drift. In *In The ICCBR'03 Workshop on Long-Lived CBR Systems* (2003), pp. 03–2003.
- [7] DELANY, S. J., CUNNINGHAM, P., AND TSYMBAL, A. A comparison of ensemble and case-base maintenance techniques for handling concept drift in spam filtering. In *Proceedings of the 19th International Conference on Artificial Intelligence (FLAIRS 2006)* (2006), G. Sutcliffe and R. Goebel, Eds., AAAI Press, pp. 340–345.
- [8] FAWCETT, T. ‘In Vivo’ spam filtering: A challenge problem for data mining. *KDD Explorations* 5, 2 (December 2003).
- [9] FDEZ-RIVEROLA, F., IGLESIAS, E., DIAZ, F., MENDEZ, J., AND CORCHADO, J. Applying lazy learning algorithms to tackle concept drift in spam filtering. *Expert Systems with Applications* 33, 1 (2007), 36 – 48.
- [10] GABER, M. M., ZASLAVSKY, A., AND KRISHNASWAMY, S. Mining data streams: a review. *SIGMOD Rec.* 34, 2 (2005), 18–26.
- [11] GOODMAN, J., AND TAU YIH, W. Online discriminative spam filter training. In *CEAS 2006: The Third Conference on Email and Anti-Spam* (2006).
- [12] HASTIE, T., TIBSHIRANI, R., AND FRIEDMAN, J. *The Elements of Statistical Learning : Data Mining, Inference and Prediction*. Springer, New York, 2001.
- [13] HSIAO, W.-F., AND CHANG, T.-M. An incremental cluster-based approach to spam filtering. *Expert Syst. Appl.* 34, 3 (2008), 1599–1608.
- [14] LOWD, D., AND MEEK, C. Adversarial learning. In *KDD '05: Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining* (New York, NY, USA, 2005), ACM, pp. 641–647.
- [15] RAYMOND, E. S., RELSON, D., ANDREE, M., AND LOUIS, G. Bogofilter. <http://bogofilter.sourceforge.net/>, 2009.
- [16] SCULLEY, D. Advances on online learning-based spam filters. In *PhD Thesis - Tufts University* (2008).
- [17] TONG, S., AND KOLLER, D. Support vector machine active learning with applications to text classification. In *Journal of Machine Learning Research* (2000), pp. 999–1006.
- [18] WIDMER, G. Learning in the presence of concept drift and hidden contexts. In *Machine Learning* (1996), pp. 69–101.
- [19] YANG, C., AND ZHOU, J. Non-stationary data sequence classification using online class priors estimation. *Pattern Recogn.* 41, 8 (2008), 2656–2664.