A Case-Based Technique for Tracking Concept Drift in Spam Filtering

Sarah Jane Delany[1], Pádraig Cunningham[2], Alexey Tsymbal[2], Lorcan Coyle[2]
[1]Dublin Institute of Technology, Kevin St., Dublin 8, Ireland.
[2]Trinity College Dublin, College Green, Dublin 2, Ireland.

Abstract

Clearly, machine learning techniques can play an important role in filtering spam email because ample training data is available to build a robust classifier. However, spam filtering is a particularly challenging task as the data distribution and concept being learned changes over time. This is a particularly awkward form of concept drift as the change is driven by spammers wishing to circumvent the spam filters. In this paper we show that lazy learning techniques are appropriate for such dynamically changing contexts. We present a case-based system for spam filtering called ECUE that can learn dynamically. We evaluate its performance as the case-base is updated with new cases. We also explore the benefit of periodically redoing the feature selection process to bring new features into play. Our evaluation shows that these two levels of model update are effective in tracking concept drift.

1 Introduction

With the cost of spam to companies worldwide estimated to be ca. $20 billion a year and growing at a rate of almost 100% a year [1], spam is a problem that needs to be handled. It is a challenging problem for a number of reasons. One of the most testing aspects is the dynamic nature of spam. Something of an arms race has emerged between spammers and the spam filters used to combat spam. As filters are adapted to contend with today's types of spam emails, the spammers alter, obfuscate and confuse filters by disguising their emails to look more like legitimate email. This dynamic nature of spam email raises a requirement for update in any filter that is to be successful over time in identifying spam.

Lazy learning is good for dynamically changing situations. With lazy learning the decision of how to generalise beyond the training data is deferred until each new instance is considered. In comparison to this, eager learning systems determine their generalisation mechanism by building a model based on the training data in advance of considering any new instances. In this paper we explore the application of Case-Based Reasoning (CBR), a lazy machine learning technique, to the problem of spam filtering and present our CBR system – Email Classification Using Examples (ECUE). We concentrate in this paper on evaluating how ECUE can assist with the concept drift that is inherent in spam.

CBR offers a number of advantages in the spam filtering domain. Spam is a disjoint concept in that spam selling cheap prescription drugs has little in common with spam offering good mortgage rates. Case-based classification works well for disjoint concepts whereas Naïve Bayes, a machine learning technique that is popular for text classification, tries to learn a unified concept description.

In addition, there is a natural hierarchy of learning available to a CBR system where the simplest level of learning is to simply update the case-base with new instances of spam or legitimate email. The advantage of CBR in this first level of learning is that it requires no rebuild of the model as is necessary with other machine learning solutions to spam filtering. The second level of learning is to retrain the system by re-selecting features that may be more predictive of spam. This level of retraining can be performed infrequently and based on newer training data. The highest level of learning, performed even more infrequently than feature selection, is to allow new feature extraction techniques to be added to the system. For instance, when domain specific features are used in the system, new feature extraction techniques will allow new features to be included. In ECUE we use a similarity retrieval algorithm based on Case Retrieval Nets (CRN) [2] which is a memory structure that allows efficient and flexible retrieval of cases. The benefit of using the CRN for implementing the second and third levels of learning is that it can easily handle cases with new features; the fact that these features may be missing on old cases is not a problem.

This paper begins in Section 2 with an overview of other work using machine learning techniques for filtering spam. Section 3 discusses the problem of concept drift and existing techniques for handling concept drift. Our case-based approach to spam filtering, ECUE, is presented in Section 4, while Section 5 presents our evaluation and results of using CBR to combat the concept drift in spam. Section 6 closes the paper with our conclusions and directions for future work.

2 Spam Filtering and Machine Learning

Existing research on using machine learning for spam filtering has focussed particularly on using Naïve Bayes [3-7]. In addition there has been work using Support Vector Machines (SVMs) [3,8,9] and Latent Semantic Indexing [10]. There has also been research using memory based classifiers [4,11]. However, this work does not address the issue of concept drift which is inherent in spam and the evaluations have been on static datasets.

One technique used for tracking concept drift is ensemble learning (see Section 3.2) which uses a set of classifiers whose individual results are combined to give an overall classification. There has been some work on ensemble learning in spam filtering using boosting [1,12] and also a combination of Naïve Bayes and memory based classifiers [13]. These evaluations use static data sets and do not attempt to track concept drift.

3 The Problem of Concept Drift

This section defines concept drift and discusses approaches to handling concept drift.

3.1 Definitions and Types of Concept Drift

A difficult problem with learning in many real-world domains is that the concept of interest may depend on some *hidden context*, not given explicitly in the form of predictive features. Typical examples include weather prediction rules that may vary radically with the season or the patterns of customers' buying preferences that may change, depending on the current day of the week, availability of alternatives, inflation rate, etc. Often the cause of change is hidden, not known in advance, making the learning task more complicated. Changes in the hidden context can induce changes in the target concept, which is generally known as *concept drift* [13]. An effective learner should be able to track such changes and to quickly adapt to them.

Two kinds of concept drift that may occur in the real world are normally distinguished in the literature: (1) sudden (abrupt, instantaneous), and (2) gradual concept drift. For example, someone graduating from college might suddenly have completely different monetary concerns, whereas a slowly wearing piece of factory equipment might cause a gradual change in the quality of output parts [15]. Stanley [15] divides gradual drift into moderate and slow drifts, depending on the rate of change.

Hidden changes in context may not only be a cause of a change in the target concept, but may also cause a change in the underlying data distribution. Even if the target concept remains the same but the data distribution changes, a model rebuild may be necessary as the model's error may no longer be acceptable. This is called *virtual concept drift* [16]. Virtual concept drift and real concept drift often occur together or virtual concept drift alone may occur, e.g. in the case of spam categorization. From a practical point of view it is not important what kind of concept drift occurs as in all cases, the current model needs to be changed.

3.2 Approaches to Handling Concept Drift

There are three approaches to handling concept drift: (1) instance selection; (2) instance weighting; and (3) ensemble learning (or learning with multiple concept descriptions). In instance selection, the goal is to select instances relevant to the current concept. The most common concept drift handling technique is based on instance selection and involves generalizing from a *window* that moves over recently seen instances and uses the learnt concepts for prediction only in the immediate future. Examples of window-based algorithms include the FLORA family of algorithms [13], FRANN [17], and Time-Windowed Forgetting (TWF) [18]. Some algorithms use a window of fixed size, while others use heuristics to adjust the window size to the current extent of concept drift, e.g. "Adaptive Size" [19] and FLORA2 [13]. Many case-base editing strategies in case-based reasoning that delete noisy, irrelevant and redundant cases are also a form of instance selection [20]. Batch

selection [19] taking groups of instances, may be considered as instance selection as well.

Instance weighting uses the ability of some learning algorithms such as SVMs to process weighted instances [19]. Instances can be weighted according to their age and their competence with regard to the current concept. Klinkenberg [19] shows in his experiments that instance weighting techniques handle concept drift worse than analogous instance selection techniques, which is probably due to overfitting the data.

Ensemble learning maintains a set of concept descriptions, predictions of which are combined using voting, weighted voting, or the most relevant description is selected. A number of techniques are used to identify the set of concept descriptions including feature construction [21], contextual clustering [22], sequential chunks of data [23,24] or groupings based on decreasing periods of time [15,25]. All incremental ensemble approaches use some criteria to dynamically delete, reactivate, or create new ensemble members, which are normally based on the base models' consistency with the current data.

Many learning algorithms were used for base models in systems handling concept drift. These include rule-based learning [13,16,21,24], decision trees, including their incremental versions [15,22-26], Naïve Bayes [24,25], SVMs [19], RBF-networks [17], and instance-based learning [18,20,27]. A problem with many global eager learners (if they are not able to update their local parts incrementally when needed) is their inability to adapt to local concept drift. In the real world, concept drift may often be local, e.g. only particular types of spam may change with time, while the others could remain the same. In the case of local concept drift, many global models are discarded simply because their accuracy on the current data falls, even if they still could be good experts in the stable parts of the data. In contrast to this, lazy learning is able to adapt well to local concept drift due to its local nature. Instance-based learning is sometimes criticized in that, as non-parametric learning, it needs relatively more instances to get high classification accuracy [13]. However, often it is not a problem in practice, as enough instances are available.

The most popular *benchmark data* for testing concept drift handling systems is represented by the STAGGER concepts [21] which are used to test most of the systems [13-17,21,22,25] or by a moving hyperplane [23-26]. A limitation of these is that they do not allow checking the algorithms' scalability to large problems, which is important as concept drift mostly occurs in big amounts of data arriving in the form of a stream. In addition, some real-world problems were used to test concept drift handling systems [22-24,26]. An important problem with most of the real-world datasets is that there is little concept drift in them, or concept drift is introduced artificially, e.g. by restricting the subset of relevant topics for each particular period of time [19].

4 A Case-Based Approach to Concept Drift

This section outlines ECUE, our case-based approach to spam filtering. It describes the feature selection, case retrieval and case-base editing techniques used.

In a CBR learner the examples in the training data are represented as cases in the case-base. In ECUE each email is a case represented as a vector of attributes or

features. Cases are set up with binary features implemented as Boolean values. If the feature exists in the email then the case assigns the feature a value of *true*, otherwise the value of the feature is *false*. It is more normal in text classification for lexical features to carry frequency information but our evaluation showed that a binary representation works better in this domain. We expect that this is due to the fact that most email messages are short and frequency information may result in overfitting.

4.1 Feature Selection

The features for each case were identified using a variety of generic lexical features, primarily by tokenising the email into words. The emails were not altered to remove HTML tags and no stop word removal, stemming or lemmatising was performed. Email attachments were removed. Since the datasets were personal it was felt that certain headers may contain useful information, so a subset of the header information was included in the tokenisation. No domain specific features were included at this stage although previous work has indicated that the efficiency of filters can be enhanced by their inclusion [6].

Tokenising 1000 emails results in a very large number of features, (tens of thousands of features). Feature selection is necessary to reduce the dimensionality of the feature space. We used Information Gain (IG) [28] to select the most predictive features as it has been shown to be an effective technique in aggressive feature removal in text classification [29]. Our cross validation experiments, varying between 100 and 1000 features across 4 datasets, indicated best performance at 700 features.

4.2 Case Retrieval

The system uses a *k*-nearest neighbour classifier to retrieve the *k* most similar cases to a target case. The standard *k*-NN algorithm individually calculates the similarity of each case in a case-base to the target case. This approach is quite inefficient in domains where there is feature-value redundancy and/or missing features in cases. Because our spam cases have both of these characteristics we use an alternative similarity retrieval algorithm based on CRNs which facilitates efficient and flexible case retrieval. The CRN is equivalent, in the cases it retrieves, to the *k*-nearest neighbour algorithm but, particularly in this situation, is more computationally efficient.

Cases are stored within the CRN as *case nodes*. A second type of node called an *Information Entity (IE) node* represents a single feature-value pair, e.g. “viagra=true” indicating the presence of the word (feature) “viagra” within a case. Case nodes are linked to the IE nodes that represent them using *relevancy arcs*. Each relevancy arc can have a weight that reflects the importance of that IE node or feature-value pair. CRNs also have the notion of *similarity arcs* between similar feature values but this is not used in ECUE as the features are all binary.

A target case activates the CRN by connecting to the IE nodes, via relevance arcs, that represent the values of its features. This activation spreads across the net to the case nodes, each of which accumulates a score appropriate to its similarity to the target case. The case nodes with the highest activations are those most similar to the target case.

Figure 1 illustrates our CRN for spam filtering. As the features in our case representation are binary, IE nodes are only included for features with a *true* value. The relevancy arcs are all weighted with a weight of 1.

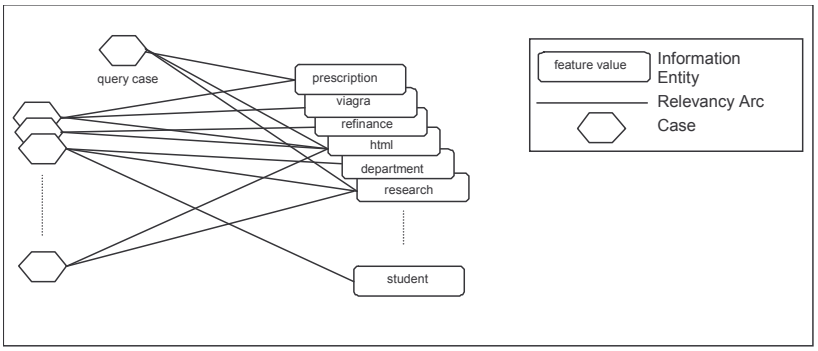

Figure 1 The CRN for spam filtering

A serious problem for spam filters is the occurrence of False Positives (FP), legitimate emails classified incorrectly as spam. An FP is significantly more serious than a False Negative (FN) (a spam email incorrectly classified as legitimate). For many people the occurrence of FPs is unacceptable. Due to this fact, the classifier used unanimous voting to determine whether the target case was spam or not. All neighbours returned had to have a classification of spam in order for the target case to be classified as spam. This strongly biases the classifier away from FPs.

4.3 Case-base Management

The approach to concept drift that we are applying is instance selection. Over time the case-base needs to be updated to include new types of spam and non-spam emails. Given the volume of emails that a single email user may get each week, there is a need to actively manage the training data. Our previous work on case-base editing techniques identified a method of case-base editing that uses the competence characteristics of a case-base to remove noisy and redundant cases. This method is called Competence-Based Editing (CBE) [30].

CBE initially builds a competence model of the case-base identifying for each case its competence properties, i.e. those cases that it contributes to classifying correctly and those it contributes to misclassifying. The technique is then a two-stage process. Using these competence properties the first stage, the competence enhancement stage, identifies and removes noisy cases. The second stage, the competence preserving stage, identifies and removes redundant cases (i.e. cases in the middle of a large cluster of cases of the same classification). The advantage of our case-editing method applied to the spam domain is that it results in a conservative pruning of the case-base which we found resulted in larger case-bases but better generalisation accuracy than typical case-editing techniques [30].

5 Evaluation

This section presents our evaluation of ECUE. Our results are presented at two levels. Firstly, we present an evaluation of the performance of the system at the first level of learning, i.e. simply updating the case-base with new examples of spam and legitimate email. Secondly, we evaluate the performance of the system when a model rebuild (i.e. when the feature selection process is redone) is carried out periodically. The evaluations were offline evaluations using emails collected over an extended period of time.

5.1 Experimental Setup

A key objective was to evaluate the performance of a CBR spam filter over a period of time to see how it handled the concept drift inherent in spam. Two datasets were used. The datasets were derived from two corpora of email, each corpus was derived from the emails sent to individuals over a period of approximately eighteen months. Dataset 1 is a collection of emails received by one individual up to and including December 2003 and Dataset 2 is a collection of emails received by a different individual up to and including January 2004. The legitimate emails include a variety of personal, business and subscribed mailing list emails. The emails were ordered in date order.

A training set of 1000 cases, 500 spam emails and 500 legitimate emails, was set up for each dataset. This training data included the last 500 spam and non spam emails received up to the end of February 2003 in the case of Dataset 1 and up to the end of January 2003 in the case of Dataset 2. This left the remainder of the data for testing. Table 1 shows the profile of the test data across each month for both datasets.

Table 1 Profile of the testing data

		Feb '03	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan '04	Tot
Data Set 1	**spam**		**629**	**314**	**216**	**925**	**917**	**1065**	**1225**	**1205**	**1830**	**576**		8902
	non spam		**93**	**228**	**102**	**89**	**50**	**71**	**145**	**103**	**85**	**105**		1076
Data Set 2	**spam**	**142**	**391**	**405**	**459**	**406**	**476**	**582**	**1849**	**1746**	**1300**	**954**	**746**	9456
	non spam	**151**	**56**	**144**	**234**	**128**	**19**	**30**	**182**	**123**	**113**	**99**	**130**	1409

A case-base was set up for each training dataset using the feature selection process described in Section 4.1 with 700 features in each case. The classifier used was *k*-nearest neighbour with $k = 3$ using unanimous voting as discussed in Section 4.2. Each email in the testing datasets, documented in Table 1, was presented for classification in date-received order to closely simulate what would happen in a real-time situation. Results were accumulated and reported at the end of each month.

5.2 Evaluation Metrics

Since an FP is more serious than an FN, accuracy (or error) as a measure of performance, does not give the full picture in that there is no transparency with regard to the numbers of FPs and FNs occurring. Two filters with similar accuracy may have very different FP and FN rates.

Previous work on spam filtering uses a variety of measures to report performance. The most common performance metrics are precision and recall [10]. Sakkis *et al.* [5] introduce a weighted accuracy measure which incorporates a measure of how much more costly an FP is than an FN. Although these measures are useful for comparison purposes, the FP and FN rate are not clear so the true effectiveness of the classifier is not evident. For these reasons we will use the rate of FPs, the rate of FNs, and the average within class error rate, *Error* = (*FPRate* + *FNRate*)/2 as our evaluation metrics. A final justification for using this set of metrics is that it is in line with how commercial spam filtering systems are evaluated on the web and in the technical press [31].

5.3 CBR vs. Naïve Bayes

As Naïve Bayes appears to be the machine learning technique of choice for spam filtering (see Section 2) we compared ECUE with a Naïve Bayes classifier [32], the results of which are summarised here.

Experiments on a number of static datasets, (subsets of those described in Section 5.1) showed that neither one of the two classifiers Naïve Bayes or the CBR classifier outperformed the other consistently [32]. These results appear to confirm other findings on static datasets [4]

Of more interest were the evaluations performed over a period of time allowing the system to dynamically update its training data with examples of spam and legitimate email that were incorrectly classified. A number of experiments were performed, varying from making no updates to the original case-base training data to updating an edited case-base on a monthly, weekly and daily basis with those emails that were misclassified over the specified period. Case-base editing used the CBE technique discussed in Section 4.3. Our evaluation showed the best performance occurred when updating an edited case-base on a daily basis with any emails misclassified that day. These results are presented in Figure 2.

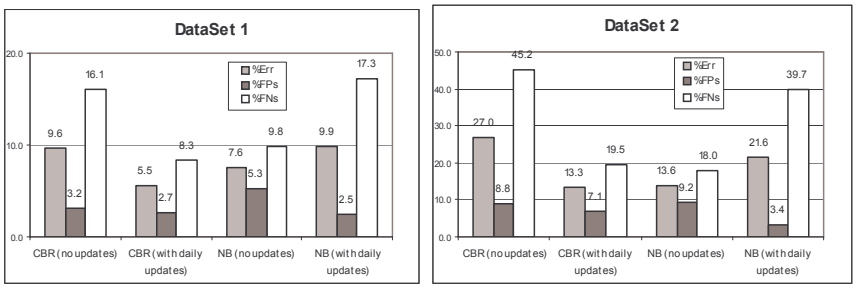

Figure 2 CBR vs. Naïve Bayes with dynamic updating

The same experiments were performed using a Naïve Bayes classifier on unedited training data. The training data could not be edited for the Naïve Bayes classifier as the editing technique is a competence-based editing technique which uses a *k*-NN classifier to determine the competence of each case in the case-base and analyses the competence properties of the cases to determine which cases should be removed. Due

to the significance of FPs, the Naïve Bayes classifier was configured to be biased away from false positives by setting the classification threshold to 1.0. Figure 2 includes the results of using Naïve Bayes.

Although Naïve Bayes has a better overall error rate over the dataset as a whole with no updating, the CBR system performs better in both datasets when dynamically updating the data to learn from incorrectly classified emails. It can be seen that daily updating of the training data with misclassified emails improves performance of the CBR system but has an overall detrimental effect on the Naïve Bayes classifier. Naïve Bayes with daily updates does improve the FP rate more significantly than ECUE but the degradation of the FN rate has an overall negative effect on performance. It is worth noting that updating a system using Naïve Bayes with any new training data requires a separate learning process to recalculate the probabilities for all features. Updating a CBR system, such as ECUE, with new training data simply requires new cases to be added to the case-base.

5.4 Level 1 Learning – Continuous Updating with new Instances

The objective of this evaluation was to examine at a detailed level the performance of the system with continuous updating of an edited case-base with the misclassified emails. The detailed results are presented in Figure 3 as (*edited cb, daily updates*). In order to illustrate the performance improvements offered by case-base editing and continuous updating, Figure 3 also includes the results of the full training case-base (*full cb, no updates*) and the edited case-base (*edited cb, no updates*) when applying all the test data in date order with no updates. Finally, for comparison purposes, Figure 3 includes a more typical window-based updating procedure (*full cb, window-based daily updates*). The original (unedited) case-base of 1000 cases was used and at the end of each day those cases that had been misclassified were added to the case-base. Then the window-based update procedure was achieved as follows; for each case that was added, remove the oldest existing case of the same classification.

The graphs in Figure 3 illustrate a number of things:

- The concept drift in the data is evident in both datasets (*full cb, no updates*).
- Although case-base editing does not show a decrease in error in Dataset 2, the decrease in FPs is significant which is crucial for this domain.
- Updating the edited case-base daily with misclassified cases shows a significant improvement in overall average error across classifications in both datasets and overall FP performance is improved in both datasets.
- Although the overall FP rate is better for window-based updates than for updates to the edited case-base for Dataset 2, it is considerably worse for Dataset 1. The average error across classifications (taking into account the FN rate) is better for updates to an edited case-base across both datasets.

An issue with continuous updating on the edited case-base is case-base management; although the editing process reduces the case-base size initially the size of the case-base grows each month. For Datasets 1 and 2 the editing process reduces the initial training case-bases to 741 and 575 cases respectively. However, the resulting size of the case-base after all the data has been applied (i.e. after 10 months for Dataset 1 and 12 months for Dataset 2) is 1512 and 2518 respectively. The

advantage of the window-based update procedure is that the case-base remains at 1000 cases always.

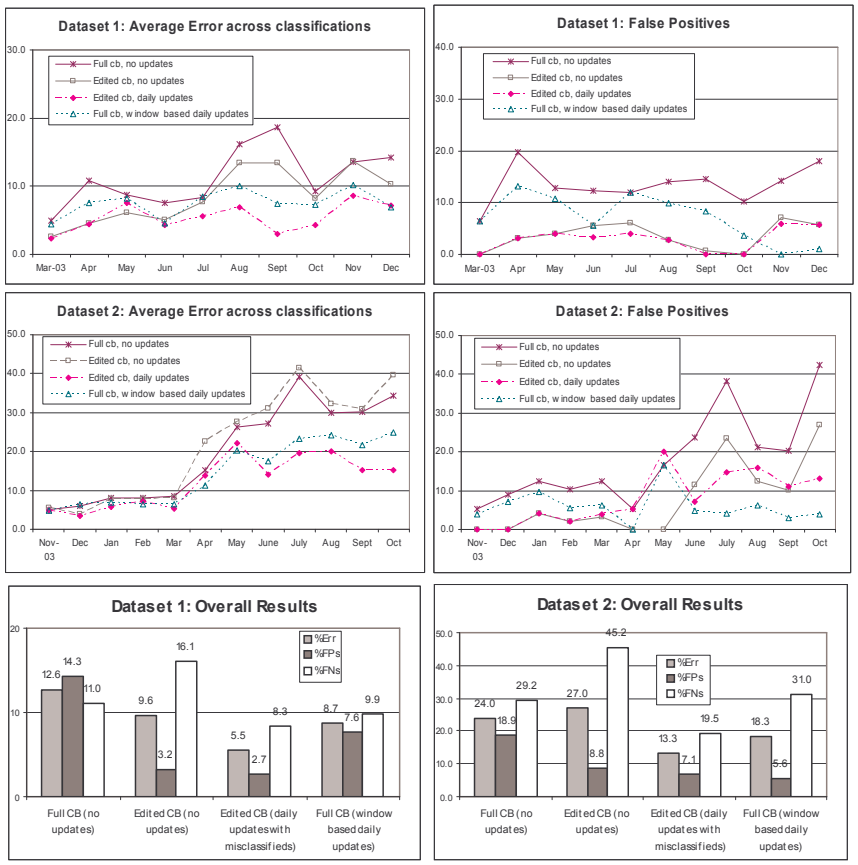

Figure 3 Results of First Level Learning - Continuous Updating

5.5 Level 2 Learning – Model Rebuild with Feature Reselection

Our next experiments involved evaluating whether the next level of learning (periodically reselecting features) improved the performance gains that we had achieved by continuous updating with misclassified cases. This level of learning involves a complete model rebuild. We chose to reselect features at the end of every 3-month period. At the reselect date, a new training set of 1000 emails was constructed. This training set consisted of the last 500 spam and 500 non-spam emails received up to the reselect date. Features were selected from this training set using the same feature selection process described in Section 4.1 above. A case-base was built

from this training set and was edited using CBE. Emails for the next 3 months were classified against this new case-base.

Figure 4 presents the results of applying feature reselection to both daily updates on an edited case-base and to window-based daily updates.

The graphs in Figure 4 illustrate the following results:

- Incorporating a periodic feature reselection and model rebuild reduces the average error across classifications for both daily update procedures.
- Applying daily updates to an edited case-base and including 3 monthly feature reselection results in lower error rates than using a daily window-based update procedure with feature reselect.
- Although the error rates reduce in all cases with feature reselection, there is no consistent improvement in FP rate for either update technique.

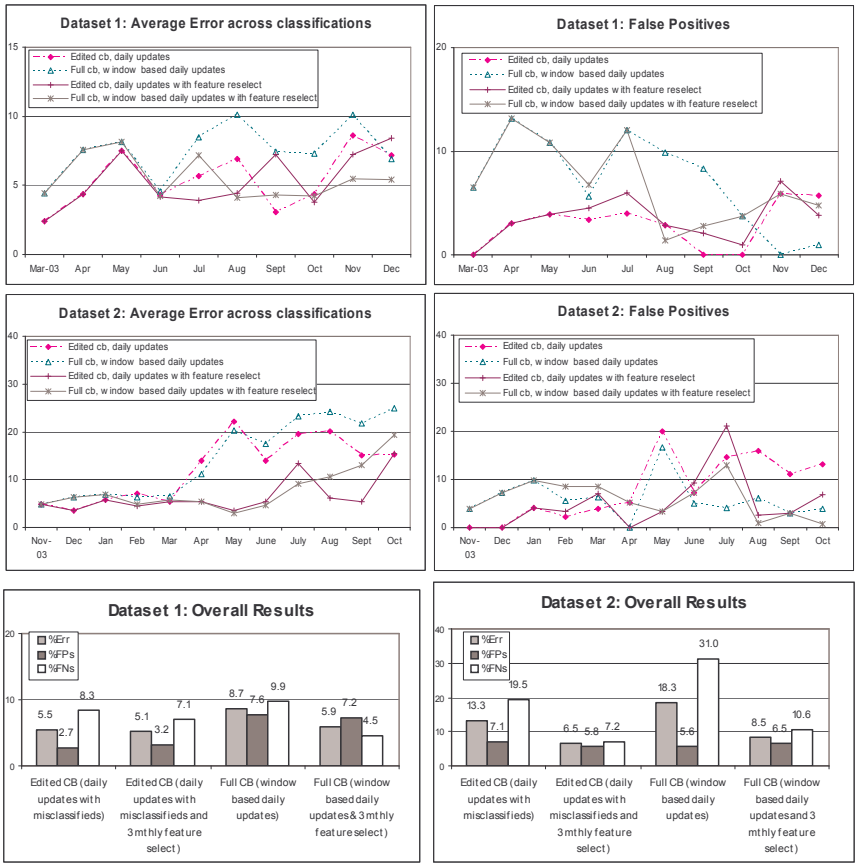

Figure 4 Results of Second Level of Learning – Feature Reselection

An advantage of including the feature reselection and case-base editing procedure for daily updates is that it controls the size of the case-base. Table 2 shows the size of

the case-base after each feature selection and case-editing process occurred on the two datasets. In all but one situation, the edited case-base has not increased over the 3-month period beyond the size of the original case-base (1000 cases). Thus, the frequency with which feature reselection is performed can be configured to manage the size of the case-base.

Table 2 Case-base sizes during evaluation with feature reselection

Dataset 1		**Dataset 2**	
Case-base size after feature select & edit	Size after 3 months of updates	Case-base size after feature select & edit	Size after 3 months of updates
741	821	575	652
718	842	628	736
678	1055	661	845
719	*798 (after 1 month)*	580	971

6 Conclusions

We have shown that using a lazy learner can handle the concept drift inherent in email spam data. Our approach is to update the case-base at the end of each day with cases that were misclassified by the system that day and to periodically rebuild the case-base using the most recent cases to reselect features. This approach performs better than the typical window-based approach at both levels of model update.

A further level of learning, to define new domain specific features is also available to our system and will be evaluated in future work. The implementation of ECUE (due to the CRN) will allow new features to be included in cases to be added to the case-base at times other than model rebuilds. However, since our model rebuild involves full feature extraction and reselection, it may not be necessary to exploit the potential of the CRN to integrate cases based on different feature sets.

Future work in this area will also include a comparison of our approach to concept drift with the more common ensemble approach to handling concept drift.

References

1. Spira J. Spam E-Mail and its Impact on IT Spending and Productivity, Basex Report 2003, http://www.basex.com/poty2003.nsfl
2. Lenz M, Auriol E, Manago M. Diagnosis and Decision Support. In: M. Bartsch-Sporl, H. D. B., and Wess, S. (eds) Case-Based Reasoning Technology: From Foundations to Applications, Springer-Verlag, 1998 LNCS 104
3. Androutsopoulos I, Paliouras G, Michelakis E. Learning to Filter Unsolicited Commercial E-Mail. Tech rpt 2004/2, 2004, NCSR "Demokritos", http://www.iit.demokritos.gr/skel/i-config/publications/
4. Androutsopoulos I, Koutsias J, Paliouras G, Karkaletsis V, Sakkis G, Spyropoulos C, Stamatopoulos, P. Learning to filter spam e-mail: A comparison of a naive Bayesian and a memory-based approach. In: 4th PKDD Workshop on Machine Learning and Textual Information Access. 2000
5. Pantel P, Lin D. SpamCop: A spam classification and organization program. In: Learning for Text Categorization—Papers from the AAAI Workshop, Madison Wisconsin, 1998, 95–98. AAAI Technical Report WS-98-05

6. Sahami M, Dumais S, Heckerman D, Horvitz E. A Bayesian Approach to Filtering Junk Email. In: AAAI-98 Workshop on Learning for Text Categorization. Madison ,Wisconsin. 1998, 55-62, AAAI Technical Report WS-98-05.
7. Androutsopoulos I, Koutsias J, Konstantinos V, Chandrinos V, Paliouras G, Spyropoulos C. An evaluation of Naive Bayesian anti-spam filtering, In: Potamias G, Moustakis V, van Someren M (eds.) Proc. of the ECML 2000 Workshop on Machine Learning in the New Information Age, 2000, 9-17
8. Drucker HD, Wu D, Vapnik V. Support vector machines for spam categorization. IEEE Transactions On Neural Networks, 1999 10(5) 1048–1054
9. Kolcz A, Alspector J. SVM-based filtering of e-mail spam with content-specific misclassification costs. In: Proc. of TextDM'2001, IEEE ICDM-2001 Workshop on Text Mining, San Jose CA 2001.
10. Gee K.R. Using Latent Semantic Indexing to Filter Spam. In: Proc. of the 2003 ACM Symposium on Applied Computing (SAC), ACM, 2003, 460-464
11. Sakkis G, Androutsopoulos I, Paliouras G, Karkaletsis V, Spyropoulos C, Stamatopoulos P. A Memory-Based Approach to Anti-Spam Filtering for Mailing Lists. Information Retrieval 2004 6(1) 49-73
12. Carreras X, Marquez L. Boosting trees for anti-spam email filtering. In: Proc. 4th Int. Conf. on Recent Advances in Natural Language Processing 2001 Tzigov Chark, Bulgaria.
13. Sakkis G, Androutsopoulos I, Paliouras G, Karkaletsis V, Spyropoulos C, Stamatopoulos P. Stacking classifiers for anti-spam filtering of e-mail. In: (ed) Lee & Harman , Proc. of 6th Conf. on Empirical Methods in Natural Language Processing 2001, 44-50
14. Widmer G, Kubat M. Learning in the presence of concept drift and hidden contexts, Machine Learning 1996 23 (1) 69-101
15. Stanley K.O. Learning concept drift with a committee of decision trees, Tech. Report UT-AI-TR-03-302, Dept of Computer Sciences, University of Texas at Austin, USA, 2003
16. Widmer G, Kubat M. Effective learning in dynamic environments by explicit context tracking, In: Proc. ECML 1993, Springer-Verlag, LNCS 667, 1993, 227-243
17. Kubat M, Widmer G. Adapting to drift in continuous domains, Tech. Report ÖFAI-TR-94-27, Austrian Research Institute for Artificial Intelligence, Vienna, 1994
18. Salganicoff M. Tolerating concept and sampling shift in lazy learning using prediction error context switching, AI Review, Spec. Iss. on Lazy Learning, 1997 11 (1-5) 133-155
19. Klinkenberg R. Learning drifting concepts: example selection vs. example weighting. Intelligent Data Analysis, Special Issue on Incremental Learning Systems Capable of Dealing with Concept Drift, 2004 8 (3) (to appear)
20. Cunningham P, Nowlan N, Delany SJ, Haahr M. A Case-Based Approach to Spam Filtering that Can Track Concept Drift. The ICCBR'03 Workshop on Long-Lived CBR Systems, Trondheim, Norway, 2003
21. Schlimmer JC, Granger RH. Incremental learning from noisy data, Machine Learning, 1986 1(3):317-354
22. Harries M., Sammut C., Horn K., Extracting hidden context, Machine Learning, 32(2), 1998, 101-126.
23. Street W, Kim Y. A streaming ensemble algorithm (SEA) for large-scale classification, Proc. 7th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining KDD-2001, ACM Press, 2001, 377-382
24. Wang H, Fan W, Yu PS, Han J. Mining concept-drifting data streams using ensemble classifiers. In: Proc. 9th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining KDD-2003, ACM Press, 2003, 226-235
25. Kolter JZ, Maloof MA. Dynamic weighted majority: a new ensemble method for tracking concept drift. In: Procs. 3rd IEEE Int. Conf. on Data Mining, IEEE CS Press, 2003, 123-130

26. Hulten G, Spencer L, Domingos P. Mining time-changing data streams. In: Proc. 7th Int. Conf. on Knowledge Discovery and Data Mining, ACM Press, 2001, 97-106.
27. Aha DW, Kibler D, Albert MK. Instance-Based Learning Algorithms. Machine Learning, 1991 6:37-66
28. Quinlan J Ross. C4.5 Programs for Machine Learning. Morgan Kaufmann Publishers, San Mateo, CA, 1993.
29. Yang Y, Pedersen JO. A comparative study on feature selection in text categorization. In: Proceedings of ICML-97, 1997, 412–420
30. Delany SJ, Cunningham P. An Analysis of Case-Based Editing in a Spam Filtering System, In: Proc. of 7th European Conf. in Case-Based Reasoning, ECCBR-04, Springer Verlag, 2004 (to appear)
31. http://www.brightmail.com/accuracy.html
32. Delany SJ, Cunningham P, Coyle L. An Assessment of Case-base Reasoning for Spam Filtering. In: Working papers of 15th Artificial Intelligence and Cognitive Science Conference (AICS 2004), 2004
33. Lewis D, Ringuette M. Comparison of two learning algorithms for text categorization, In: SDAIR, (1994) 81-93.
34. Niblett. Constructing decision trees in noisy domains. In: Proceedings of the Second European Working Session on Learning, Sigma, 1987, 67-78.
35. Kohavi R, Becker B, Sommerfield D. Improving Simple Bayes. In: ECML-97 Proceedings of the Ninth European Conference on Machine Learning. 1997