Artif Intell Rev
DOI 10.1007/s10462-007-9041-6

Textual case-based reasoning for spam filtering: a comparison of feature-based and feature-free approaches

Sarah Jane Delany · Derek Bridge

Received: 1 October 2006 / Revised: 30 October 2006

Abstract Spam filtering is a text classification task to which Case-Based Reasoning (CBR) has been successfully applied. We describe the ECUE system, which classifies emails using a feature-based form of textual CBR. Then, we describe an alternative way to compute the distances between cases in a feature-free fashion, using a distance measure based on text compression. This distance measure has the advantages of having no set-up costs and being resilient to concept drift. We report an empirical comparison, which shows the feature-free approach to be more accurate than the feature-based system. These results are fairly robust over different compression algorithms in that we find that the accuracy when using a Lempel-Ziv compressor (GZip) is approximately the same as when using a statistical compressor (PPM). We note, however, that the feature-free systems take much longer to classify emails than the feature-based system. Improvements in the classification time of both kinds of systems can be obtained by applying case base editing algorithms, which aim to remove noisy and redundant cases from a case base while maintaining, or even improving, generalisation accuracy. We report empirical results using the Competence-Based Editing (CBE) technique. We show that CBE removes more cases when we use the distance measure based on text compression (without significant changes in generalisation accuracy) than it does when we use the feature-based approach.

Keywords Spam filtering · Case-based reasoning · Case-base editing · Case-based maintenance · Feature selection · Distance measures · Text compression

S. J. Delany
Dublin Institute of Technology, Dublin, Ireland
e-mail: sarahjane.delany@comp.dit.ie

D. Bridge (✉)
University College Cork, Cork, Ireland
e-mail: d.bridge@cs.ucc.ie

1 Introduction

Spam email has proved to be a problem that is enduring and difficult to solve. In January 2004, Bill Gates predicted that spam email would be eradicated as a problem within 2 years.[1] The fact that this prediction did not come to pass demonstrates the severity of the problem. Identifying spam is a difficult task for a number of reasons. Spam is a diverse concept: spam advertising cheap prescription drugs has little in common with spam offering investment opportunities. In addition, spam is constantly changing, new opportunities are persistently being exploited by spammers and seasonal effects such as advertising weight loss products after Christmas have an impact. In the case of spam, however, the main factor that gives rise to what machine learning research calls concept drift is that spammers continually change the content and structure of spam email in order to bypass the mechanisms in place to stop them. There is also a subjective and personal aspect to identifying spam: what is considered to be spam by one individual may not be considered spam by others. Finally, mistakingly identifying a legitimate email as spam (known as a False Positive) is very significant in this domain and is unacceptable to most email users.

Of the wide range of strategies that have been used to combat spam some of the more effective have been: whitelists and blacklists,[2] authentication-based techniques,[3] and spam filtering including both collaborative filters (Gray and Haahr 2004) and content-based filters. In this paper we focus on ECUE, a personalised, content-based filter that uses Case-Based Reasoning (CBR) to classify emails. ECUE has been shown to be successful at filtering spam (Delany et al. 2005a) and at handling concept drift in spam (Delany et al. 2005b).

The case representation used in ECUE is feature-based. In this paper we describe an alternative way to calculate the distances between cases which is feature-free, using a distance measure based on text compression. The feature-free distance measure performs considerably better than the feature-based measure and has the advantage of having no set-up costs and being resilient to concept drift.

The remainder of the paper is organised as follows. Section 2 outlines ECUE, the spam filtering system used for the evaluation in this paper. Section 3 discusses the feature-based approach to spam filtering, identifying how features are extracted, selected and represented. Section 4 then discusses the feature-free alternative approach, describing the compression-based distance measure that can be used in textual CBR. Evaluations of the accuracies and classification times of both the feature-based and feature-free approaches are described in Sect. 5. Section 6 describes a case base editing technique called Competence-Based Editing and evaluates the technique for both the feature-based and feature-free approaches. The paper concludes in Sect. 7 with a discussion of the results and an outline of possible future work.

2 Email classification using examples

ECUE (Delany et al. 2005a, b) is a personalised case-based machine learning system that uses past examples of a user's email as training instances. A case base of examples of an individual's previously received emails, both spam and legitimate, is set up. New emails are classified against the case base using the k-Nearest Neighbour (k-NN) algorithm. The k cases that are the nearest neighbours i.e. the closest in distance, to the target case are returned and

[1] http://www.theregister.com/2004/01/26/well_kill_spam_in_two/

[2] www.email-policy.com/Spam-black-lists.htm

[3] www.emailauthentication.org/

used to generate a classification for the target case. Due to the significance of False Positives (FPs), the classification process uses unanimous voting to bias the classifier away from FP classifications. This requires all k neighbours retrieved by the k-NN algorithm to be of class *spam* before the target case can be classified as spam. A Case Retrieval Net (Lenz et al. 1998) is used to speed up the retrieval process.

An advantage of the case-based approach to spam filtering is that it can easily learn incrementally. Any email that is misclassified can be inserted into the case base, along with its correct classification, in the hope that this improves the competence of the system.

But one of the challenges of using CBR for spam filtering is to manage the training data, choosing those training examples that are best at prediction. Prior to classification, case base editing can be performed on the case base to reduce the number of cases. Our case base editing technique is described and evaluated in Sect. 6.

3 Feature-based textual CBR

Each training instance in ECUE is a case e_j represented as a vector of feature values, $e_j = (f_{1j}, f_{2j}, \dots f_{nj}, \mathit{class})$ with *class* representing the class of the email, either *spam* or *nonspam*.

The features are identified by lexical analysis of the textual content of the email. No stopword removal or stemming is performed on the text. Email attachments are removed but any HTML text present in the email is included. As ECUE is a personalised filter, the header fields may contain useful information and a selection of header fields, including the *Subject*, *To* and *From* headers, are included in the tokenisation.

Three types of features are extracted: word features, character features and structural features (e.g. the proportion of uppercase characters, lowercase characters or white space in the email). The *feature extraction* process results in a large number of features. In addition, the representation of each email is sparse, with only a small number of the total feature set having a value other than zero. *Feature selection* using Information Gain (IG) (Quinlan 1997) is performed to identify the features which are most predictive of spam or legitimate mails. Based on the results of preliminary cross-validation experiments, we chose to use 700 features for the evaluations in this paper.

The case representation we use for each email is binary; if the feature exists in the email the feature value $f_{ij} = 1$, otherwise $f_{ij} = 0$. It is more normal in text classification to use numeric-valued features (e.g. occurrence frequencies). But the results of evaluations showed that for this domain the implications of the higher classification time and higher case base editing time when using numeric features outweighed the minor improvement in accuracy achieved by using numeric features, especially considering that there were no significant improvements in the rate of FPs (Delany et al. 2005a). These experiments also show that, from the point of view of the FP rate in particular, it is better not to use *feature weighting* on the binary representation.

To obtain the binary representation for word features, we use a simple existence rule: the feature is set to 1 if and only if the word appears in the email. For character features almost all characters will occur within an email so the existence rule is not useful. Instead, for character features the IG value (used above for selecting features) is also used as a threshold to indicate whether the feature should be set in the case representation or not. Specifically, if the normalised frequency of the character in the email is greater than or equal to the normalised frequency which returns the highest information gain for that character, then the feature is set to 1. The same rule is applied to structural features: the values are initially proportions between zero and one, but these are thresholded using the IG to give a binary representation.

Given that we are using a binary representation, the distance between target case e_t and a case from the case base e_c is simply a count of features on which they disagree:

$$FDM(e_t, e_c) =_{\text{def}} \sum_{i=1}^{n} |f_{it} - f_{ic}| \tag{1}$$

By using a Case Retrieval Net to store the case base, *FDM* can be computed extremely efficiently.

4 Feature-free textual CBR

There is a feature-free alternative to feature-based textual CBR. As we will explain, we can define a distance measure based on text compression (Li et al. 2003). Distance measures based on data compression have a long history in bioinformatics, where they have been used, e.g., for DNA sequence classification (Loewenstern et al. 1995). Outside of bioinformatics, compression-based distance measures have been applied to *clustering* of time-series data (Keogh et al. 2004) and languages (Benedetto et al. 2002; Cilibrasi and Vitanyi 2005). They have also been applied to *classification* of time series data (Keogh et al. 2004). But, to the best of our knowledge, they have not been applied to text categorisation in general or spam filtering in particular.[4] There have, however, been other classifiers based on text compression. In these classifiers, for each class an adaptive statistical compressor builds a compression model from training examples belonging to that class. The classifier assigns a target document to the class whose compression model best accounts for that document (Frank et al. 2000; Teahan 2000; Teahan and Harper 2001). Bratko *et al.* have recently used classifiers of this kind for spam filtering (Bratko and Filipič 2005; Bratko et al. 2006). Rennie and Jaakkola (2002), on the other hand, propose using text compression to discover features indicative of spam 2002.

Koegh et al. (2004) and Li et al. (2003) have both presented generic distance measures based on data compression and inspired by the theory of Kolmogorov complexity. The *Kolmogorov complexity* $K(x)$ of a string x can be defined as the size of the smallest Turing machine capable (without any input) of outputting x to its tape. The *conditional Kolmogorov complexity* $K(x|y)$ of x relative to y can be defined as the size of the smallest Turing machine capable of outputting x when given y as an input. This can be the basis of a distance measure. Informally, if $K(x|y) < K(x|z)$, then y contains more information content that is useful to outputting x than z does, and so y is more similar to x than z is.

One possible way to define a normalised distance measure using Kolmogorov complexity is:

$$d_K(x, y) =_{\text{def}} \frac{K(x|y) + K(y|x)}{K(xy)} \tag{2}$$

where $K(xy)$ is the size of the smallest Turing machine for outputting y concatenated to x.

Unfortunately, Kolmogorov complexity is not computable in general, and so we must approximate it. Since the Kolmogorov complexity of a string is in some sense the size of the smallest description of the string, one way of thinking of $K(x)$ is that it is the length of the

[4] But see the discussion of the possibility of using compression in instance-based classification of email at www.kuro5hin.org/story/2003/1/25/224415/367

best compression we can achieve for x. So, we can approximate $K(x)$ by $C(x)$, the size of x after compression by a data compressor. Then distance can be defined as

$$d_C(x, y) =_{\text{def}} \frac{C(x|y) + C(y|x)}{C(xy)} \tag{3}$$

where $C(x|y)$ is the size of x after compression by a compressor that has first been 'trained' on y, $C(y|x)$ is defined analogously, and $C(xy)$ is the compressed size of y concatenated to x.

But d_C also has problems because standard compressors do not allow easy computation of $C(x|y)$. Hence, following Keogh *et al.* (2004), we make a further simplification. Given strings x and y, a Compression-based Dissimilarity Measure (*CDM*) can be defined as follows:

$$CDM(x, y) =_{\text{def}} \frac{C(xy)}{C(x) + C(y)} \tag{4}$$

The intuition here is that compression of xy will exploit not only the redundancies within x and within y but also inter-document redundancies (similarities) between x and y too. If there are inter-document redundancies, then the amount of compression of xy should be greater than we obtain by compressing x and y separately.

CDM produces values in the range (0.5, 1]. Even with the best possible compression algorithm, the lowest value it can produce is slightly above 0.5 because, even if $x = y$, $C(xy)$ will be slightly greater than $C(x)$. In principle *CDM*'s maximum value is 1. This would occur when x and y are so different that $C(xy) = C(x) + C(y)$ and so compressing y within xy is not helped by having compressed x first.

It should also be noted that properties expected of distance measures do not hold. In general, it is not the case that $CDM(x, x) = 0$ iff $x = y$; $CDM(x, y) \neq CDM(y, x)$, i.e. *CDM* is not symmetric; and $CDM(x, y) + CDM(y, z) \not\geq CDM(x, z)$, i.e. the triangle-inequality does not hold. None of this prevents use of *CDM* in, for example, classification tasks, provided the classification algorithm does not rely on any of these properties. For example, an exhaustive implementation of k-NN (in which the algorithm finds the k nearest neighbours to the query by computing the distance between the query and *every* case in the case base) will work correctly. But retrieval algorithms that rely on these properties to avoid computing some distances (e.g. k–d trees (Wess et al. 1994) and Fish and Shrink (Schaaf 1996)) are not guaranteed to work correctly.

CDM is a feature-free approach to computing distance. Cases are represented by raw text: there is no need to extract, select or weight features; there is no need to tokenise or parse queries or cases. *CDM* works directly on the raw text. We discuss the advantages of this in Sect. 7.

5 Spam filtering experiments

We conducted an experimental evaluation whose objective was to replace the feature-based distance measure that ECUE uses with a compression-based distance measure and to compare the two measures.

The datasets used in this evaluation were derived from two corpora of email. Each email corpus is a personal collection of the spam and legitimate email received by an individual over a period of approximately two years. The legitimate emails in each corpus include a variety of personal, business and mailing list emails. Two datasets (Datasets 1.1 and 1.2) were extracted from one corpus, while Datasets 2.1 and 2.2 were extracted from the other.

Each dataset consists of 1000 emails, 500 of each class, received over a period of approximately three months. Most individuals do not receive equal volumes of spam and legitimate email, but the actual distributions vary considerably from person to person. Weiss and Provost (2003) conclude that using a balanced distribution is a reasonable default when the true distribution is not available, and this is what we have chosen here.

Since FP classifications are significant in this domain, straightforward classification accuracy (or error) as a measure of performance does not give the full picture. The evaluation metrics we use include:

(i) The error rate, i.e. the overall proportion of emails that were not filtered correctly (*%Err*).

(ii) The FN rate, i.e. the proportion of spam emails that were missed (*%FNs*).

(iii) The FP rate, i.e. the proportion of legitimate emails that were classified as spam (*%FPs*).

We compare the performance of the different classifiers by calculating confidence levels using McNemar's test (Salzberg 1997).

Figure 1 compares the feature-based distance measure (*FDM*) with the compression-based distance measure (*CDM*) on each of the four datasets, using $k = 3$. In this figure, GZip is the compressor used to compute *CDM*. The graphs include the results we obtain for unedited case bases ("full CB") and those we obtain for edited case bases produced by Competence-Based Editing ("edited CB"). These latter results are not discussed until Sect. 6.

As can be seen in the figure, using *CDM*, the feature-free compression-based approach, gives much better accuracy: the overall error (*%Err*) between *CDM*(full CB) and *FDM*(full CB) is significant in all cases at the 99.9% level. The FP rate is also lower for *CDM* than for *FDM* in all datasets except for Dataset 2.1. However, the differences in FP rate for both Datasets 2.1 and 2.2 is not significant.

Figure 2 compares the use of two different compression algorithms in *CDM*, GZip and PPM. GZip is a variant of Lempel-Ziv compression, in which a repetition of a string within a text may be replaced by a pointer to an earlier occurrence. In GZip, substitutions are confined to a 32 Kb sliding window. PPM, Prediction by Partial Matching (Cleary and Witten 1984), is an adaptive statistical compressor. A statistical compressor builds a probabilistic model from which it can predict the most likely next character in the stream, and encodes the more probable characters in fewer bits. If the model is of order n, then the next character is predicted based on the previous n characters. An adaptive compressor updates its model on the basis of the character frequencies seen so far, hence the bit pattern used to encode a character may change. PPM adaptively builds models of all orders up to n; it uses the model with largest order, but if a novel character is encountered, an escape symbol is included in the output and PPM switches to the model with next lowest order. A default model at level -1 ensures that every character can be encoded. In our experiments, we tried orders of 2, 4 and 8.[5] Orders above 6 or so generally do not increase the amount of compression (Cleary and Witten 1984).

In general, PPM is thought to achieve some of the best compression rates. However, on the emails in our corpora we found GZip to be slightly better: its average compression was 59% compared with 53.3% for PPM(2), 56.6% for PPM(4) and 56.9% for PPM(8). There is not much difference between the compression rates achieved for spam and non-spam emails, with approximately 0.5% difference either way. The better the compression rate, the closer $C(x)$ will approximate $K(x)$, the Kolmogorov complexity, which can be thought of as the

[5] In these experiments, we use Bob Carpenter's implementation of PPM: http://www.colloquial.com/ArithmeticCoding/

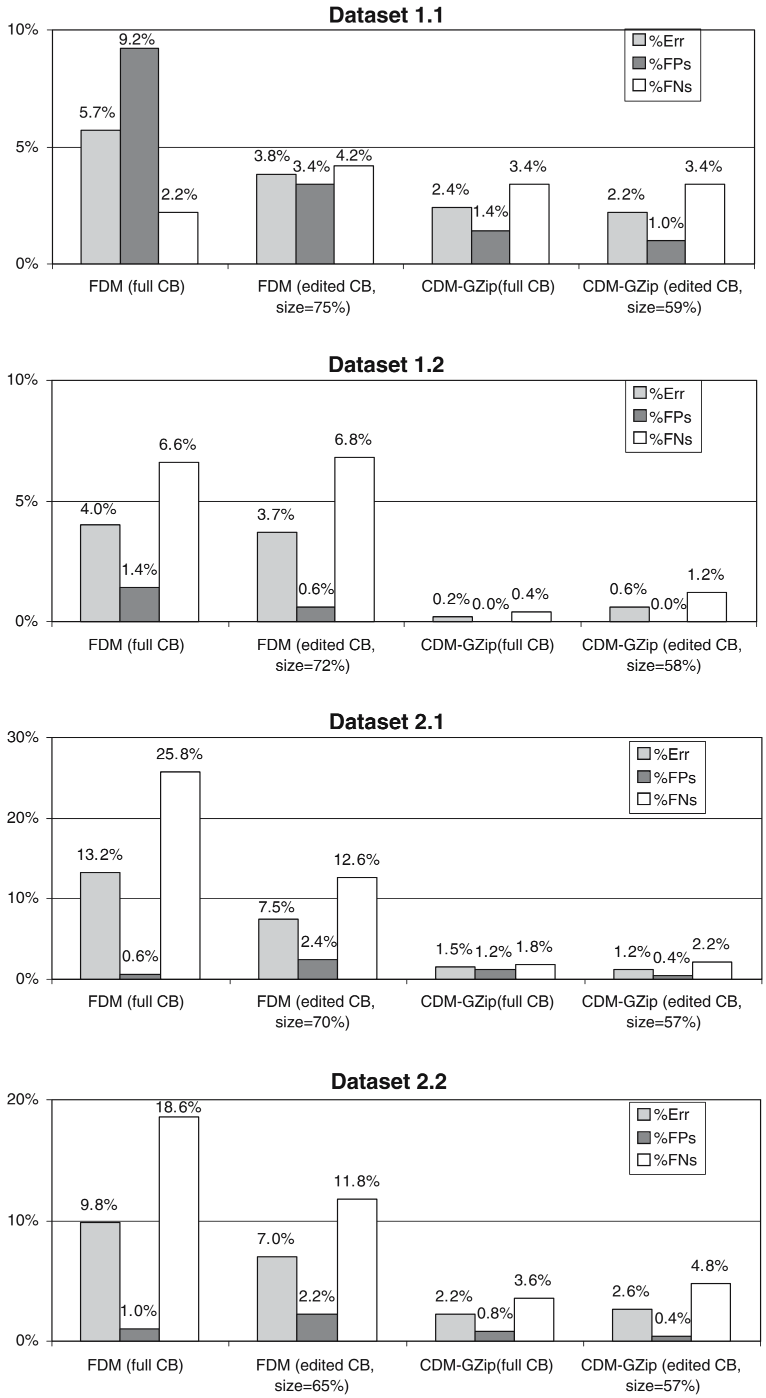

Fig. 1 Comparison between feature-based (*FDM*) and compression-based (*CDM*) distance, using 10-fold cross validation on each dataset

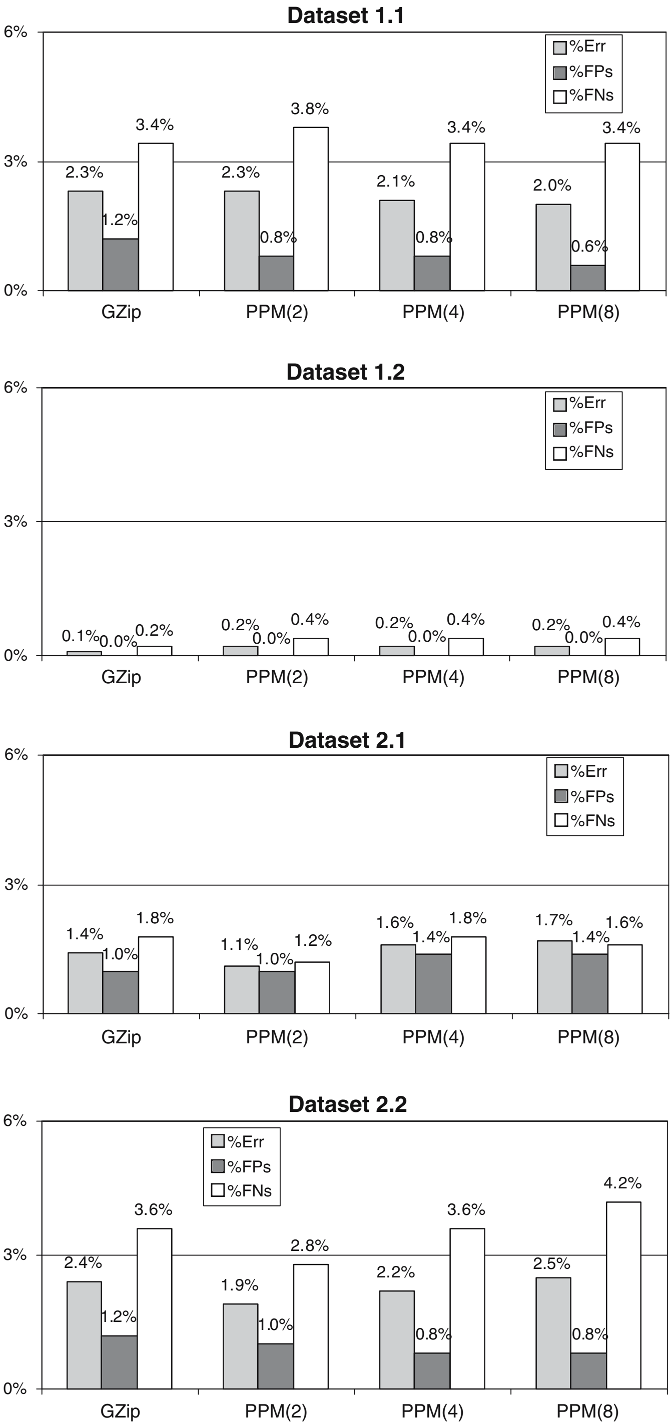

Fig. 2 Comparison between GZip compression and PPM compression for *CDM*, using leave-one-out cross validation on each dataset

best compression one can achieve on x. But this does not mean using the better compressor in *CDM* will result in a better approximation of d_K, the distance measure based on Kolmogorov complexity. This is because the improvement in compression rate of the better of two compressors on the different terms in Eq. 4 may not be the same (Cilibrasi and Vitanyi 2005).

In fact, the results in Fig. 2 show that there is little difference in classification error between the different compression algorithms. None of the differences is statistically significant using McNemar's measure. This suggests that the choice of compression algorithm does not matter greatly and supports the findings in (Cilibrasi and Vitanyi 2005), where results on *clustering* tasks were fairly robust over different compressors.

A limitation of the compression-based approach is the time it takes to classify an email. Table 1 shows the time taken in seconds to classify a single email using feature-based distance and feature-free distance with a case base of 1,000 cases. The time to classify a single email using *CDM* is, at best, 180 times slower than using the feature-based distance. The compression algorithm is computationally much more expensive than comparing feature values. Furthermore, compression-based distance requires the target case to be compared with each case in the case base, which is not always necessary in feature-based similiarity if, for example, a Case Retrieval Net is used.

Using *CDM* with GZip performs significantly better in terms of computation time than *CDM* with PPM. *CDM* with GZip can also be made somewhat faster by modifying the length of the email files to take into account the fact that the GZip algorithm uses a sliding window size of 32 Kb. Truncating the email files to 16 Kbytes each before calculating the *CDM* achieves speed-ups of between 9.5%, and 25% on the datasets evaluated. The *CDM*-GZip figures in Table 1 include this speed up. We found that the truncation of the email files does not have any real effect on the classification error results. The results in Fig. 1 also include this speed-up.

6 Competence-based editing

Case base editing algorithms remove redundant or noisy cases from a case base, thus reducing retrieval time, while endeavouring to maintain or even improve the generalisation accuracy. There is significant research in this area, e.g. (Smyth and Keane 1995; McKenna and Smyth 2000; Wilson and Martinez 2000; Brighton and Mellish 2002). The case base editing technique that we use is called Competence-Based Editing (CBE) (Delany and Cunningham 2004). CBE builds a competence model of the case base by identifying for each case its usefulness (represented by the cases that it contributes to classifying correctly) and also the damage that it causes (represented by the cases that it causes to be misclassified). These properties of each case are used in a two step process to identify the cases to be removed. The first step is the competence enhancement or noise reduction step, which removes noisy cases

Table 1 Time to classify one email in seconds using different distance measures on the unedited case bases

Dataset	Feature-based	*CDM*-Gzip	*CDM*-PPMZ(2)	*CDM*-PPMZ(4)	*CDM*-PPMZ(8)
Dataset 1.1	0.01	2.00	30.8	37.2	69.2
Dataset 1.2	0.01	1.84	24.9	28.3	32.7
Dataset 2.1	0.01	1.82	24.2	28.1	32.4
Dataset 2.2	0.01	1.97	25.5	29.8	34.7

that adversely affect classification accuracy. The second step is the competence preservation or redundancy reduction step, which removes redundant cases that are not needed for correct classification. CBE has been shown to conservatively reduce the size of a spam case base while maintaining and even improving its generalisation accuracy (Delany and Cunningham 2004).

The name Blame-Based Noise Reduction (BBNR) is used to refer to the technique that CBE uses in its noise reduction step. For each case in the case base, BBNR makes use of two sets, the coverage and liability sets. The coverage set for case *c* contains the cases that have *c* as one of their *k* nearest neighbours and which have the same class as *c*. The liability set contains the cases that have *c* as one of their *k* nearest neighbours but which have a different class from *c*. BBNR looks at all cases in the case base that might cause another case to be misclassified (i.e. it looks at all cases that have non-empty liability sets). For each case *c* with a non-empty liability set (taken in descending order of the size of their liability sets), if the cases in *c*'s coverage set can still be classified correctly without *c* then *c* can be deleted. Hence, unlike most approaches, which remove cases that are misclassified by other cases, BBNR emphasises the removal of cases that cause misclassifications of other cases.

The name Conservative Redundancy Reduction (CRR) is used to refer to the technique that CBE uses in its redundancy reduction step. CRR makes further use of the coverage sets of the cases in the case base. In CRR, cases are taken from the existing case base in ascending order of the size of their coverage sets and are transferred to an initially empty new case base. For each case *c* that is transferred from the existing case base, the cases that *c* can be used to classify (in other words, *c*'s coverage set) are removed from the existing case base and so they will not be transferred to the new case base. By taking cases in ascending order of coverage set size, CRR transfers cases that are close to class boundaries first. This makes it more conservative and less detrimental to generalisation accuracy than approaches that take the cases in the reverse order; they consider first the cases that are in the centre of clusters of cases having the same class.

We used Competence-Based Editing (comprising BBNR followed by CRR) to edit the case bases used in our experiments using both *FDM* (the feature-based distance measure) on the one hand and *CDM* (the feature-free compression-based distance measure) on the other hand. The results are included in Fig. 1.

The generalisation accuracy of the four datasets edited using *CDM* is higher than those edited using *FDM*, significant at the 95% level or higher in all cases. In addition to this, the FP rate is lower for all datasets, significant at the 95% level or higher in all cases except for Dataset 1.2 where the FP rate is already 0% for both the full and the edited case base using *CDM*. This agrees with Delany and Cunningham's (2004) conclusions that case base editing improves the FP rate.

CBE is conservative in removing cases in the spam domain, producing larger edited case bases than other editing algorithms but with the best generalisation accuracy (Delany and Cunningham 2004). It is interesting to note from Fig. 1 that, while the reductions are still relatively conservative compared with other case editing techniques (see results in (Delany and Cunningham 2004)), the resulting size of the case base after editing using *CDM* is smaller than that produced by editing using *FDM*. It is also worth noting that editing using *CDM* removes different cases from the case base from those removed when editing using *FDM*. This suggests that there are some shortcomings in the bag of words representation of documents typically used in feature-based textual CBR.

7 Conclusion

In this paper we have shown that a feature-free approach to spam filtering, such as that offered by the compression-based distance measure, has several advantages over a feature-based approach. The first is its remarkable accuracy: our experiments show significantly higher classification accuracy than the normal feature-based distance measure typically used in textual CBR.

A second advantage is its low set-up costs: the raw text files are used directly and so feature extraction, selection and weighting are all unnecessary. This is a major advantage when one considers that spam is a personal and diverse concept.

A third advantage, related to the second, concerns concept drift. Delany et al. (2005a) describe a three-level hierarchy of actions for coping with concept drift that would be needed in a production-quality feature-based spam filter. Level 1 is regular case base update, i.e. updating the case base with misclassified emails. Level 2, with lower frequency, is feature selection, i.e. periodically reselecting features from the most recently selected set of candidate features. Level 3, with lowest frequency, is feature extraction, i.e. periodically re-extracting a set of candidate features from the most recent training examples. A feature-free approach requires only Level 1 actions. This is a major advantage when one considers how constantly spam changes.

The feature-free approach also has its disadvantages. These are at least twofold. First, *CDM* returns only a number, denoting distance. It does not return any factors that could be used to *explain* its judgements or to drive case adaptation. Adaptation is not relevant to spam filtering and it traditionally has lower importance in textual CBR, but lack of explainability could inhibit broader up-take.

A second disadvantage is computation time. Computing distances using *CDM* is slower than using *FDM*. *CDM*'s computation time varies with the compression algorithm used, e.g. GZip is faster than PPM; GZip also allows for a speed-up (based on its window size), which was not detrimental to accuracy in our domain. Our experiments show that on an unedited case base of 1,000 emails *CDM* with GZip takes up to 2 s to classify an email compared to 0.01 s for the feature-based system. Rarely will a user ever notice this cost: it is unlikely to matter if an email is available for viewing an extra 2 s after delivery.

The classification time (of all the classifiers) will be lower if the case base is smaller. In live experiments with ECUE, case base sizes were significantly smaller than 1,000, at approximately 300 cases (Delany et al. 2006). In this paper we have reported the results of editing the case bases using Competence-Based Editing. We obtained much greater reductions in case base size when using *CDM* (with no significant changes in overall error rate or FP rate) than when using *FDM*. This would suggest that *CDM*'s higher classification time need not be an impediment to broader up-take.

There may be other ways of further reducing the number of distance computations. For example, the compression-based distance measure described in (Li et al. 2003) comes close enough to satisfying the conventional properties expected of distance measures (triangle inequality, etc.) to allow use of retrieval algorithms (such as *k*–*d* trees and Fish and Shrink) that rely on these properties. Precomputation and caching of $C(x)$ for each case x as it enters the case base will also help.

Finally, we should consider how robust a spam filter that uses *CDM* might be. Spammers are constantly trying to outwit the latest spam filters. How easy will they find it to outwit a compression-based approach? One possibility, which spammers use even now, is to place all content into images, rendering it inaccessible to filters that look only at the textual content. Another possibility, which is also in current use, is to add large quantities of spam salad, i.e.

random text, to the end of the message. To outwit spam filters, spammers must ensure that the spam salad they insert into different spam emails is adequately dissimilar. The extent to which spammers can use this idea to outwit the feature-free *CDM* approach is not yet clear.

Future work in this area will include further work on the computation time issues, investigating algorithms to speed up retrieval time. We will also perform an empirical investigation of *CDM*'s resilience to the concept drift in spam. We will extend the application of *CDM* to texts other than emails, to tasks other than classification, and to text other than raw text, e.g. text that has undergone POS-tagging.

Acknowledgements The authors are grateful to Pádraig Cunningham for discussions about this work.

References

Benedetto D, Caglioti E, Loreto V (2002) Language trees and zipping. Phys Rev Lett 88(4):048702

Bratko A, Cormack GV, Filipič B, Lynam TR, Zupan B (2006) Spam filtering using statistical data compression models. J Mach Learn Res 7:2673–2698

Bratko A, Filipič B (2005) Spam filtering using character-level Markov models: Experiments for the TREC 2005 spam track. In: Proceedings of the 14th text retrieval conference. Gaithersburg, MD

Brighton H, Mellish C (2002) Advances in Instance Selection for Instance-Based Learning Algorithms. Data Min Knowl Disc 6(2):153–172

Cilibrasi R, Vitanyi P (2005) Clustering by compression. IEEE Trans Infor Theory 51(4):1523–1545

Cleary JG, Witten IH (1984) Data compression using adaptive coding and partial string matching. IEEE Trans Commun 32(4):396–402

Delany S, Cunningham P, Coyle L (2005) An assessment of case-based reasoning for spam filtering. Artifi Int Rev 24(3–4):359–378

Delany SJ, Cunningham P (2004) An analysis of case-based editing in a spam filtering system. In: Funk P, González-Calero P (eds) 7th European conference on case-based reasoning (ECCBR 2004), vol 3155 of LNAI. Springer, pp 128–141

Delany SJ, Cunningham P, Symth B (to appear) ECUE: A spam filter that uses machine learning to track concept drift. In: Proceedings of the 17th European Conference on artificial intelligence (PAIS stream)

Delany SJ, Cunningham P, Tsymbal A, Coyle L (2005) A case-based technique for tracking concept drift in spam filtering. Knowl-Based Syst 18(4–5):187–195

Frank E, Chui C, Witten IH (2000) Text categorization using compression models. In: Proceedings. of the IEEE data compression conference. Utah, USA, pp 200–209

Gray A, Haahr M (2004) Personalised, collaborative spam filtering. In: Proceedings of 1st conference on email and anti-spam

Keogh E, Lonardi S, Ratanamahatana C (2004) Towards parameter-free data mining. In: KDD '04, Proceedings of the 10th ACM SIGKDD, International conference on knowledge discovery and data mining. New York, NY, USA, pp 206–215, ACM Press

Lenz M, Auriol E, Manago M (1998) Diagnosis and decision support. In: Lenz M, Bartsch-Sporl B, Burkhard H, Wess S (eds) Case-based reasoning technology, from foundations to applications. Springer-Verlag, pp 51–90

Li M, Chen X, Li X, Ma B, Vitanyi P (2003) The similarity metric. In: Proceedings of the 14th annual ACM-SIAM symposium on discrete algorithms. pp 863–872

Loewenstern D, Hirsh H, Yianilos P, Noordewier M (1995) DNA sequence classification using compression-based induction. Technical Report 95-04, DIMACS

McKenna E, Smyth B (2000) Competence-guided case-base editing techniques. In: Proceedings of the 5th European workshop on CBR. pp 186–197

Quinlan JR (1997) C4.5 Programs for machine learning. Morgan Kaufmann Publishers Inc, San Mateo, CA

Rennie JDM, Jaakkola T (2002) Automatic feature induction for text classification. In: MIT artificial intelligence laboratory Abstract Book. Cambridge, MA

Salzberg S (1997) On comparing classifiers: Pitfalls to avoid and a recommended approach. Data Min Knowl Disc 1:317–327

Schaaf, JW (1996) Fish and shrink. A next step towards efficient case retrieval in large-scale case bases. In: Smith I, Faltings B (eds) Advances in case-based reasoning. pp 362–376

Smyth B, Keane M (1995) Remembering to forget: A competence-preserving deletion policy for case-based reasoning. In: Proceedings of the 14th international joint conference on AI. pp 377–383

Teahan, WJ (2000) Text classification and segmentation using minimum cross-entropy. In: Proceedings of the 6th international conference on recherche d'information assistee par ordinateur. Paris, France, pp 943–961

Teahan WJ, Harper DJ (2001) Using compression-based language models for text categorization. In: Proceedings of the workshop on language modeling for information retrieval. Carnegie Mellon University, pp 83–88

Weiss G, Provost F (2003) Learning when training data are costly: the effect of class distribution on tree induction. J Artif Int Res 19:315–354

Wess S, Althoff K-D, Derwand G (1994) Using *k–d* trees to improve the retrieval step in case-based reasoning. In: Wess S, Althoff K-D, Richter M (eds) Topics in case-based reasoning. Springer, pp 167–181

Wilson DR, Martinez TR (2000) Reduction techniques for instance-based learning algorithms. Mach Learn 38:257–286