Proceedings of Student/Faculty Research Day, CSIS, Pace University, May 7th, 2004

A Neural Network Classifier for Junk E-Mail

Ian Stuart, Sung-Hyuk Cha, and Charles Tappert

Abstract. Most e-mail readers spend a non-trivial amount of time regularly deleting junk e-mail (spam) messages, even as an expanding volume of such e-mail occupies server storage space and consumes network bandwidth. An ongoing challenge, therefore, rests within the development and refinement of automatic classifiers that can distinguish legitimate e-mail from spam. A few published studies have examined spam detectors using Naïve Bayesian approaches and large feature sets of binary attributes that determine the existence of common keywords in spam, and many commercial applications also use Naïve Bayesian techniques. Spammers recognize these attempts to thwart their messages and have developed tactics to circumvent these filters, but these evasive tactics are themselves patterns that human readers can often identify quickly. This preliminary study tests an alternative approach using a neural network (NN) classifier on a corpus of e-mail messages from one user. The feature set uses descriptive characteristics of words and messages similar to those that a human reader would use to identify spam. The results of this study are compared to previous spam detectors that have used Naïve Bayesian classifiers.

1 Introduction

The volume of junk e-mail (spam) transmitted by the Internet has arguably reached epidemic proportions. While the inconvenience of spam is not new – public comments about unwanted e-mail messages identified the problem as early as 1975 – the volume of unsolicited commercial e-mail was relatively limited until the mid-1990s [3]. Spam volume was estimated to be merely 8% of network e-mail traffic in 2001 but has ballooned to about 40% of e-mail today. One research firm has predicted that the cost of fighting spam across the U.S. will approach $10 billion in 2003 [12].

Most e-mail readers must spend a non-trivial amount of time regularly deleting spam messages, even as an expanding volume of junk e-mail occupies server storage space and consumes network bandwidth. An ongoing challenge, therefore, rests within the development and refinement of automatic classifiers that can distinguish legitimate e-mail from spam.

Many commercial and open-source products exist to accommodate the growing need for spam classifiers, and a variety of techniques have been developed and applied toward the problem, both at the network and user levels. The simplest and most common approaches are to use filters that screen messages based upon the presence of common words or phrases common to junk e-mail. Other simplistic approaches include *blacklisting* (automatic rejection of messages received from the addresses of known spammers) and *whitelisting* (automatic acceptance of message received from known and trusted correspondents). In practice, effective spam filtering uses a combination of these three techniques. The primary flaw in the first two approaches is that it relies upon complacence by the spammers by assuming that they are not likely to change (or forge) their identities or to alter the style and vocabulary of their sales pitches. Whitelisting risks the possibility that the recipient will miss legitimate e-mail from a known or expected correspondent with a heretofore unknown address, such as correspondence from a long-lost friend, or a purchase confirmation pertaining to a transaction with an online retailer.

A variety of text classifiers have been investigated that categorize documents topically or thematically, including probabilistic, decision tree, rule-based, example-based ("lazy learner"), linear discriminant analysis, regression, support vector machine, and neural network approaches [10]. A prototype system has also been designed to recognize hostile messages ("flames") within online communications [11]. However, the body of published academic work specific to spam filtering and classification is limited. This may seem surprising given the obvious need for effective, automated classifiers, but it suggests two likely reasons for the low volume of published material. First, the effectiveness of any given anti-spam technique can be seriously compromised by the public revelation of the technique since spammers are aggressive and adaptable. Second, recent variations of Naïve Bayesian classifiers have demonstrated high degrees of success. In general, these classifiers identify attributes (usually keywords or phrases common to spam) that are assigned probabilities by the classifier. The product of

the probabilities of each attribute within a message is compared to a predefined threshold, and the messages with products exceeding the threshold are classified as spam.

Sahami, et al. [9] proposed a Naïve Bayesian approach that examined manually-categorized messages for a set of common words, phrases ("be over 21", "only $", etc.), and non-textual characteristics (such as the time of initial transmission or the existence of attachments) deemed common to junk e-mail. Androutsopoulos, et al. [1] used an edited,[1] encrypted, and manually-categorized corpus of messages with a lemmatizer and a stop-list, using words-attributes. Both approaches used binary attributes, where $X_n = 1$ if a property is represented and $X_n = 0$ if it is not. In each case, the selected words were the result of hand-crafted, manually-derived selections. In addition to these approaches, several applied solutions exist that claim high success rates (as high as 99.5%) with Naïve Bayesian classifiers [2], [5], [6], [7], [8] that use comprehensive hash tables comprised of hundreds of thousands of tokens and their corresponding probability values, essentially creating attribute sets of indefinite size.[2]

While these Naïve Bayesian approaches generally perform effectively, they suffer from two intrinsic problems. The first is that they rely upon a consistent vocabulary by the spammers. New words that become more frequently used must be identified as they appear in waves of new spam, and, in the case of hash tables, any new word must be assigned an initial arbitrary probability value when it is created. Spammers use this flaw to their advantage, peppering spam with strings of random characters to slip the junk messages under the classification thresholds. The second problem is one of context. Binary word-attributes, and even phrase-attributes, do not identify the common patterns in spam that humans can easily and readily identify, such as unusual spellings, images and hyperlinks, and patterns typically hidden from the recipient, such as HTML comments.

In summary, Naïve Bayesian classifiers are indeed naïve, and require substantial calculations for each e-mail classification. A human reader, by contrast, requires relatively little calculation to deduce if a given e-mail is a legitimate message or spam. While spammers send messages that vary widely in composition, subject, and style, they typically include identifiable tactics that are designed to garner attention or to circumvent filters and classifiers and that are rarely used in traditional private correspondence. These evasive tactics are themselves patterns that human readers can often identify quickly.

In this paper we apply a neural network (NN) approach to the classification of spam using attributes comprised from descriptive characteristics of the evasive patterns that spammers employ, rather than the context or frequency of keywords in the messages. This approach produces similar results but with fewer attributes than the Naïve Bayesian strategies.

2 Methodology

This project used a corpus of 1654 e-mails received by one of the authors over a period of several months. None of the e-mails contained embedded attachments.

Each e-mail message was saved as a text file, and then parsed to identify each header element (such as `Received:` or `Subject:`) to distinguish them from the body of the message. Every substring within the subject header and the message body that was delimited by white space was considered to be a *token*, and an *alphabetic word* was defined as a token delimited by white space that contains only English alphabetic characters (A-Z, a-z) or apostrophes. The tokens were evaluated to create a set of 17 hand-crafted features from each e-mail message (Table 1).

---

[1] The corpus utilized by [1] removed all HTML tags and attachments, and all header fields other than "Subject:" were removed for privacy reasons.

[2] A token is a "word" separated by some predetermined delimiter (spaces, punctuation, HTML tags, etc.), and therefore a given token many not necessarily correspond to an actual word of written text. Examples from [5] include "qvp0045", "freeyankeedom", "unsecured", and "7c266675", among others. In [6], Graham argues that performance may be improved by providing separate case-sensitive entries for words in a hash table (such as "FREE!!!" and "Free!!!"), potentially magnifying the size of the probability table. The selection of delimiters and the effectiveness of scanning HTML tags for tokens are currently subjects of debate.

Table 1. Features extracted from each e-mail.

Feature	Features From the Message Subject Header
1	Number of alphabetic words that did not contain any vowels
2	Number of alphabetic words that contained at least two of the following letters (upper or lower case): J, K, Q, X, Z
3	Number of alphabetic words that were at least 15 characters long
4	Number of tokens that contained non-English characters, special characters such as punctuation, or numeric digits at the beginning or middle of the token.
5	Number of words with all alphabetic characters in upper case
6	Binary feature indicating occurrence of a character (including spaces) that is repeated at least three times in succession: yes = 1, no = 0
	Features From the Priority and Content-Type Headers
7	Binary feature indicating whether a priority header appeared within the message headers (`X-Priority` and/or `X-MSMail-priority`) or whether the priority had been set to any level besides normal or medium: yes = 1, no = 0
8	Binary feature indicating whether a `content-type` header appeared within the message headers or whether the content type of the message has been set to "text/html": yes = 1, no = 0
	Features From the Message Body
9	Proportion (fraction) of alphabetic words with no vowels and at least seven characters
10	Proportion of alphabetic words that contained at least two of the following letters in upper or lower case: J, K, Q, X, Z
11	Proportion of alphabetic words that were at least 15 characters long
12	Binary feature indicating whether the white-space-delimited strings "From:" and "To:" were both present: 1 = yes, 0 = no
13	Number of HTML opening comment tags
14	Number of hyperlinks ("href=")
15	Number of clickable images represented in the HTML
16	Binary feature indicating whether a color of any text within the body message was set to white: 1 = yes, 0 = no
17	Number of URLs within hyperlinks that contain any numeric digits or any of three special characters ("&", "%" or "@") in the domain or subdomain(s) of the link

The e-mails were manually categorized into 800 legitimate e-mails and 854 junk e-mails. Half of each category was randomly selected to comprise a training set (n = 827) and the remaining e-mails were used as a testing set. All feature values were scaled (normalized) to range from 0 to 1.

The training data were used to train a three-layer, backpropagation neural network with the number of hidden nodes ranging from 4 to 14 and the number of epochs from 100 to 500. After training, the e-mail messages of the testing set were classified to obtain generalization accuracy results.

3 Results

The relative success of spam filtering techniques is determined by classic measures of precision and recall on the testing subsets of legitimate e-mail and junk e-mail. Spam precision (SP) is defined as the percentage of messages classified as spam that actually are spam. Likewise, legitimate precision (LP) is the percentage of messages classified as legitimate that are indeed legitimate. Spam recall (SR) is defined as the proportion of the number of correctly-classified spam messages to the number of messages originally categorized as spam. Similarly, legitimate recall (LR) is the proportion of correctly-classified legitimate messages to the number of messages originally categorized as legitimate. Thus, we define the counts:

n_{SS} = the number of spam messages correctly classified as spam.
n_{SL} = the number of spam messages incorrectly classified as legitimate
n_{LL} = the number of legitimate messages correctly classified as legitimate
n_{LS} = the number of legitimate messages incorrectly classified as spam

and the precision and recall formulas:

$$Spam\ \ precision\ (SP) = \frac{n_{SS}}{n_{SS} + n_{LS}} \quad (1)$$

$$Legitimate\ \ precision\ (LP) = \frac{n_{LL}}{n_{LL} + n_{SL}} \quad (2)$$

$$Spam\ \ recall\ (SR) = \frac{n_{SS}}{n_{SS} + n_{SL}} \quad (3)$$

$$Legitimate\ \ recall\ (LR) = \frac{n_{LL}}{n_{LL} + n_{LS}} \quad (4)$$

Table 2 gives the results on the testing set by hidden node count and training epochs. The trial with 12 hidden nodes and 500 epochs (highlighted in the table) produced the lowest number of misclassifications, with 35 of the 427 spam messages (8.20%) classified as legitimate (n_{SL}), and 32 of the 400 legitimate messages (8.00%) classified as spam (n_{LS}), for a total of 67 misclassifications.

Table 2. Classification results on the testing set (n = 827).

Hidden Nodes	Training Epochs	Spam		Legitimate	
		Precision (%)	Recall (%)	Precision (%)	Recall (%)
8	300	91.81	86.65	86.56	91.75
	400	90.95	89.46	88.94	90.50
	500	93.73	87.59	87.62	93.75
10	300	92.11	90.16	89.73	91.75
	400	91.09	86.18	86.05	91.00
	500	92.48	86.42	86.45	92.50
12	300	93.52	87.82	87.79	93.50
	400	91.73	88.29	87.98	91.50
	500	**92.45**	**91.80**	**91.32**	**92.00**
14	300	91.58	84.07	84.37	91.75
	400	92.04	86.65	86.59	92.00
	500	91.28	88.29	87.92	91.00

Of the 35 misclassified spam messages, 30 were short in length – only a few lines, including HTML tags – some as brief as "save up to 27% on gas" followed by a hyperlink. Among the remaining five messages: one had many "comments" without comment delimiters, thus creating nonsense HTML tags that some browsers ignore (but some do not – a risk this spammer was willing to take); two were written almost entirely in ASCII

escape codes; one followed four image files with English words in jumbled, meaningless sentences; and one creatively used an off-white color for fonts to disguise the random characters appended to the end of the e-mail.

The 32 legitimate messages were misclassified due mostly to characteristics that are unusual for personal e-mail. Twenty-two affected the features normally triggered by spam: six were from a known correspondent that prefers to write in white typeface on a colored background, ten were responses or forwards that quoted HTML that triggered several features, five were commercial e-mail from known vendors (with many hyperlinks and linkable images), and one was ranked as "low" priority from a known correspondent. The remaining ten messages were less obvious: four included special characters or vowel-less words in the subject header, three had several words with multiple occurrences of rare English characters (feature 2), and three had an unusual number of hyperlinks (due, in part, to links in signature lines).

The NN accuracy of this study is similar to that of the Naïve Bayesian classifiers described in [1] and [9], and Table 3 presents a comparison.

Table 3. Comparison results for NN and Naïve Bayesian classifiers.

Classifier	Num Feat	Num Msgs	Spam (%)	SP (%)	SR (%)	LP (%)	LR (%)
NN (12 nodes, 500 epochs)	17	827	51.6	92.5	91.8	91.3	92.0
Naïve Bayesian from [9][3]							
Words	500	1789	88.2	97.1	94.3	87.7	93.4
Words+Phrases	500	1789	88.2	97.6	94.3	87.8	94.7
Words+Phrases+Non-textual	500	1789	88.2	100.0	98.3	96.2	100.0
Naïve Bayesian from [1][4]							
Bare	50	1099	43.8	95.1	84.0	N/A	N/A
Stop-List	50	1099	43.8	96.8	84.2	N/A	N/A
Lemmatized	100	1099	43.8	98.3	78.1	N/A	N/A
Lemmatized + Stop List	100	1099	43.8	98.0	79.6	N/A	N/A

For comparison purposes we also ran a small experiment with spam blacklist databases. While some databases are part of commercial programs, most require manual entry of IP addresses one at a time, apparently designed primarily for mail server administrators who are trying to determine whether their legitimate e-mails are being incorrectly tagged as spam. To test how accurately legitimate and spam e-mails are tagged by the blacklist databases, we manually entered the IP addresses of the e-mail messages that were incorrectly tagged by the NN classifier (32 legitimate and 35 spam e-mails) into a site that sends IP addresses to 173 working spam blacklists and returns the number of hits [4]. We entered both the first (original) IP address of each message and also, when present, a second IP address (a possible mail server or ISP). While it is likely that the second IP column includes bulk e-mail servers of spammers, it is also likely that it includes non-spamming ISPs or Web portals that route junk e-mail messages but presumably do not participate intentionally in spamming. Because we considered single-list hits to be anomalies since they aren't confirmed by any other blacklists on the site, we counted only hit counts greater than one as spam that would have been blacklisted. The blacklisting results are presented in Table 4. While the percentages of legitimate e-mails considered spam by the blacklists are lower than the percentages of spam correctly identified as spam, it is surprising to see that over half were incorrectly screened using our "at least two blacklists" criterion. Even though we tested the blacklist databases with potentially difficult e-mails, the ones incorrectly classified by the NN classifier, the poor blacklisting results indicate that the blacklisting strategy, at least for these databases, is inadequate.

Table 4. Blacklisting results for the e-mails incorrectly tagged by the NN classifier.

[3] Sahami, et. al. [9] used three feature sets in their approach. The first used keywords, the second considered additional key phrases, and the last included non-textual attributes. In each case the most prevalent 500 attributes within the corpus were selected.

[4] Androutsopoulos, et. al. [1] used a corpus from a moderated mailing list in a "bare" form and with three forms of alterations: a lemmatized version (which changed parts of speech, such as changing "earned" to "earn"), a version edited with a stop-list (which removed frequently used words), and a version using both the lemmatizer and a stop-list. The authors did not provide statistics for legitimate precision (LP) or legitimate recall (LR).

Classification	Blacklisting (% Considered Spam)		
	First IP Address (Original Address)	Second IP Address (E-mail Server/ISP)	Either First or Second IP Address
n_{LS} (32 E-mails)	53.1	25.0	53.1
n_{SL} (35 E-mails)	40.0	60.0	97.1

4 Conclusion

Although the NN technique is accurate and useful, its spam precision performance is not high enough for it to be used without supervision. For this technique to be more useful, the feature set would require additional members or modifications. It should be noted, however, that the NN required fewer features to achieve results similar to the Naïve Bayesian approaches, indicating that descriptive qualities of words and messages, similar to those used by human readers, can be used effectively to distinguish spam by a classifier. As suggested in previous work [9], a combination of keywords and descriptive characteristics may provide more accurate classification. A neural network classifier using these descriptive features, however, may not degrade over time as rapidly as classifiers that rely upon a relatively static vocabulary from spammers. Strategies that apply a combination of techniques, such as a NN with a whitelist, would likely yield better results.

References

1. Androutsopoulos, I; Koutsias, J; Chandrinos, K. V. and Spyropoulos, C. D. 2000. An Experimental Comparison of Naive Bayesian and Keyword-Based Anti-Spam Filtering with Personal E-mail Messages. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Athens, Greece, 2000), pp. 160-167.
2. Burton, B. 2002. SpamProbe – Bayesian Spam Filtering Tweaks. http://spamprobe.sourceforge.net/paper.html; last accessed November 17, 2003.
3. Cranor, L. F. and LaMacchia, B. A. 1998. Spam! *Communications of the ACM,* 41(8): pp. 74-83.
4. Declude, IP Lookup Against a List of All Known DNS-based Spam Databases. http://www.declude.com/junkmail/support/ip4r.htm; last accessed January 27, 2004.
5. Graham, P. 2002. A Plan for Spam. http://www.paulgraham.com/spam.html; last accessed November 17, 2003.
6. Graham, P. 2003. Better Bayesian Filtering. *In Proceedings of the 2003 Spam Conference* (Cambridge, Massachusetts, 2003). See *http://spamconference.org/proceedings2003.html*.
7. Hauser, S. 2002. Statistical Spam Filter Works for Me. http://www.sofbot.com/article/Statistical_spam_filter.html; last accessed November 17, 2003.
8. Hauser. S. 2003. Statistical Spam Filter Review. http://www.sofbot.com/article/Spam_review.html; last accessed November 17, 2003.
9. Sahami, M.; Dumais, S.; Heckerman, D. and Horvitz, E. 1998. A Bayesian Approach to Filtering Junk E-mail. In *Learning for Text Categorization—Papers from the AAAI Workshop* (Madison, Wisconsin, 1998), AAAI Technical Report WS-98-05, pp. 55-62.
10. Sebastiani, F. 2002. Machine Learning in Automatic Text Categorization. *ACM Computing Surveys (CSUR),* 34(1): pp. 1-47.
11. Spertus, E. 1997. Smokey: Automatic Recognition of Hostile Messages. In *Proceedings of the 14th National Conference on AI and the 9th Conference on Innovative Applications of AI* (Providence, Rhode Island, 1997), pp. 1058-1065.
12. Weiss, A. 2003. Ending Spam's Free Ride. *netWorker*, 7(2): pp. 18-24.