Unsupervised Spam Detection based on String Alienness Measures

Kazuyuki Narisawa
Department of Informatics
Kyushu University
Motooka 744
Fukuoka 819-0395, Japan
k-nari@i.kyushu-u.ac.jp

Hideo Bannai
Department of Informatics
Kyushu University
Motooka 744
Fukuoka 819-0395, Japan
bannai@i.kyushu-u.ac.jp

Kohei Hatano
Department of Informatics
Kyushu University
Motooka 744
Fukuoka 819-0395, Japan
hatano@i.kyushu-u.ac.jp

Masayuki Takeda
Department of Informatics
Kyushu University
Motooka 744
Fukuoka 819-0395, Japan
SORST, Japan Science and Technology Agency
takeda@i.kyushu-u.ac.jp

ABSTRACT

We propose an unsupervised method for detecting spam documents from Web page data, based on *equivalence relation*s on strings. We propose 3 measures for quantifying the *alienness* (i.e. how different it is from others) of substring equivalence classes within a given set of strings. A document is then classified as spam if it contains a characteristic equivalence class as a substring. The proposed method is unsupervised, independent of language, and is very efficient. Computational experiments conducted on data collected from Japanese web forums show fairly good results.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Retrieval models; I.5.4 [**Applications**]: Text processing

General Terms

Algorithm, Experimentation, Performance

Keywords

Spam Detection, Equivalence Class

1. INTRODUCTION

Due to its remarkable development, the Web has become a major means of advertisement [6]. Not only normal websites, but CGM (Consumer Generated Media) which is made and written by the casual user, is also exploited as an advertisement media. *Spam* messages, which are unsolicited, unwanted advertisement messages that are sent or posted by spammers, is becoming a huge issue on this media, because in general, any user can freely and easily post messages.

There exist various types of spam called wikispam (spam in Wikis), splog (spam in Weblogs), commentspam (spam in forums), spam mail (spam in email), and more recently, spim (spam over Instant Messaging) [4], and spit (spam over IP Telephony). These spams advertise their goods and Websites, mislead users to access other websites, manipulate the PageRank of their sites and so on. Not only do these messages interfere with the user trying to obtain useful information, they can overload the servers which provide various services to the users.

There are roughly three strategies for combatting spam. One is Regulation, such as the "no follow tag" [14]. This strategy does not detect spams, but tries to prevent spams from affecting the results of automated link analyses. Another is Link Analysis, which detects spams and junk mutual link sets, by link structure analysis [2, 1, 5]. Although link analysis can detect spam with high accuracy, it suffers from the drawback that it generally has a high computational cost, and that it can only be used for spam messages that contain links. The third is Contents Analysis [11], which detects and filters spams by syntactic analysis. There are various learning-based filters such as Bayesian filters [12], which are fairly effective. However, such supervised methods must first be fed with a large amount of training message data marked as spam and nonspam, which may be costly to generate.

In this paper, we consider an unsupervised method for the detection of spam in document sets, based on the *alienness* of the substrings contained in each document. In order to effectively transmit their advertisement message to their potential customers (victims), spammers send a large number of spam messages. We assume that these messages must be in some way "different" from other messages, and quantify this amount using several measures based on substring equivalence classes defined in [13]. Using these measures, we output spam documents that contain such alien substring equivalence classes.

In Section 2, we introduce some simple notations, as well as the substring amplification method [10] which is our previous unsupervised method for detecting spam. In Section 3,

WWW2007, May 8–12, 2007, Banff, Canada.
.

the new spam detection algorithm is introduced. We show results of computational experiments conducted on web forum postings in Section 4. Section 5 concludes the paper.

2. PRELIMINARIES

Let Σ be a finite alphabet. An element of Σ^* is called a *string*. Strings x, y and z are said to be a *prefix*, *substring*, and *suffix* of the string $u = xyz$, respectively, and the string u is said to be a *superstring* of substring y. The length of a string u is denoted by $|u|$. The empty string is denoted by ε, that is, $|\varepsilon| = 0$. Let $\Sigma^+ = \Sigma^* - \{\varepsilon\}$. The i-th character of a string u is denoted by $u[i]$ for $1 \leq i \leq |u|$, and the substring of u that begins at position i and ends at position j is denoted by $u[i:j]$ for $1 \leq i \leq j \leq |u|$. For convenience, let $u[i:j] = \varepsilon$ for $j < i$. The set of substrings of a string w is denoted by $Sub(w)$, and let $Sub(S) = \bigcup_{w \in S} Sub(w)$ for a set S of strings. The elements of $Sub(S)$ are called *substrings* of S. For a set S of strings $w_1, w_2, \ldots, w_l$, let $||S||$ denote the total length of strings in S, that is, $||S|| = \Sigma_{k=1}^{l} |w_k|$. Let $|S|$ denote the cardinality of S, that is, $|S| = l$. Let $Sub_f(S)$ denote the set of substrings appearing f times in S.

2.1 Substring Amplification

We introduce the Substring Amplification Method presented in [10], which is one of unsupervised spam detection methods (see, e.g., [15] for other unsupervised spam detection methods). The Substring Amplification Method is conceptually similar to the method in this paper in that it tries to detect spams by detecting deviations in substring frequencies of documents. Unlike methods such as n-gram analysis and term analysis, it first enumerates all substrings of the input documents. Then the frequency distribution of the substrings is plotted. More precisely, the frequency of substrings (the total number of times a given substring appears in the document set) is taken on the x-axis, and the number of substrings which have that frequency, i.e. $|Sub_f(S)|$, is taken on the y-axis. Figure 1 is an example plot for the Substring Amplification Method. This distribution seemingly follows the Zipf's law [17, 16]. Looking more closely at this graph, outliers from the distribution are observed to be due to substrings from spam documents.

The Substring Amplification Method produces such plots as Figure 1 (More precisely, it transforms plots in a way that outliers are exaggerated. See [10] for the details). We can detect outliers, which are possibly spams, by picking them up manually or by using the simple heuristics that detects outliers iteratively [10]. Even though the Substring Amplification Method can visualize candidates of spams very well, the automated detection heuristics are not always satisfactory. For example, Figure 2 shows the result obtained by Substring Amplification Method with the heuristics in [10] run on the web data described in Section 4 (forum 4314).

We propose a new method using an equivalence relations on substrings, which improves the Substring Amplification Method. In the next chapter, we describe our method in detail.

3. OUR METHOD

We consider the equivalence relation over substrings, introduced by Blumer et al. [3] based on their occurrences. Intuitively, each equivalence class gathers the substrings whose "occurrences" are the same. Using the representative ele-

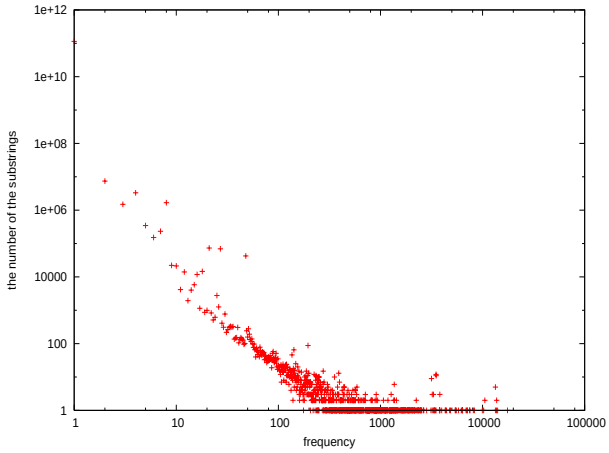

Figure 1: f-$|Sub_f(S)|$ plot with Substring Amplification Method. The distribution for nonspams seems to follow the Zipf's law, the distribution for spams does not.

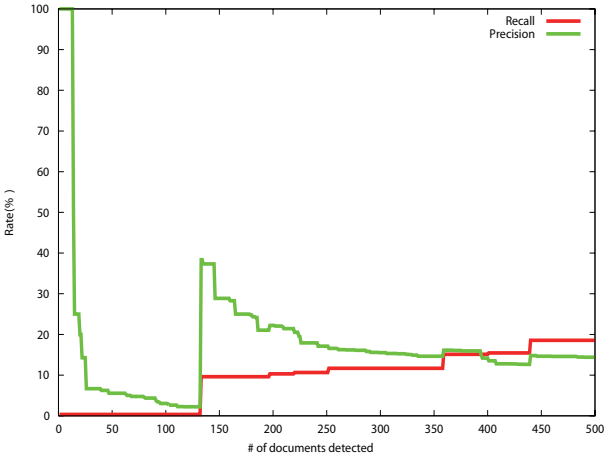

Figure 2: The result of the Substring Amplification Method with the heuristics in [10].

ments of the equivalence classes, we are able to effectively do away with the many different substrings which specify the same positions in the text.

We note that by using the suffix array data structure [8] together with its lcp array, we can enumerate the equivalence classes, as well as each of the measures that will be used in this paper, in linear time [9]. The algorithm is a non-trivial extension to the algorithm of [7], which is beyond the scope of this paper.

3.1 Equivalence relation on substrings

In this subsection, we give definitions of the equivalence relation of Blumer et al. [3], and then state some properties.

3.1.1 Definition

Definition 1. Let S be a non-empty finite subset of Σ^+.

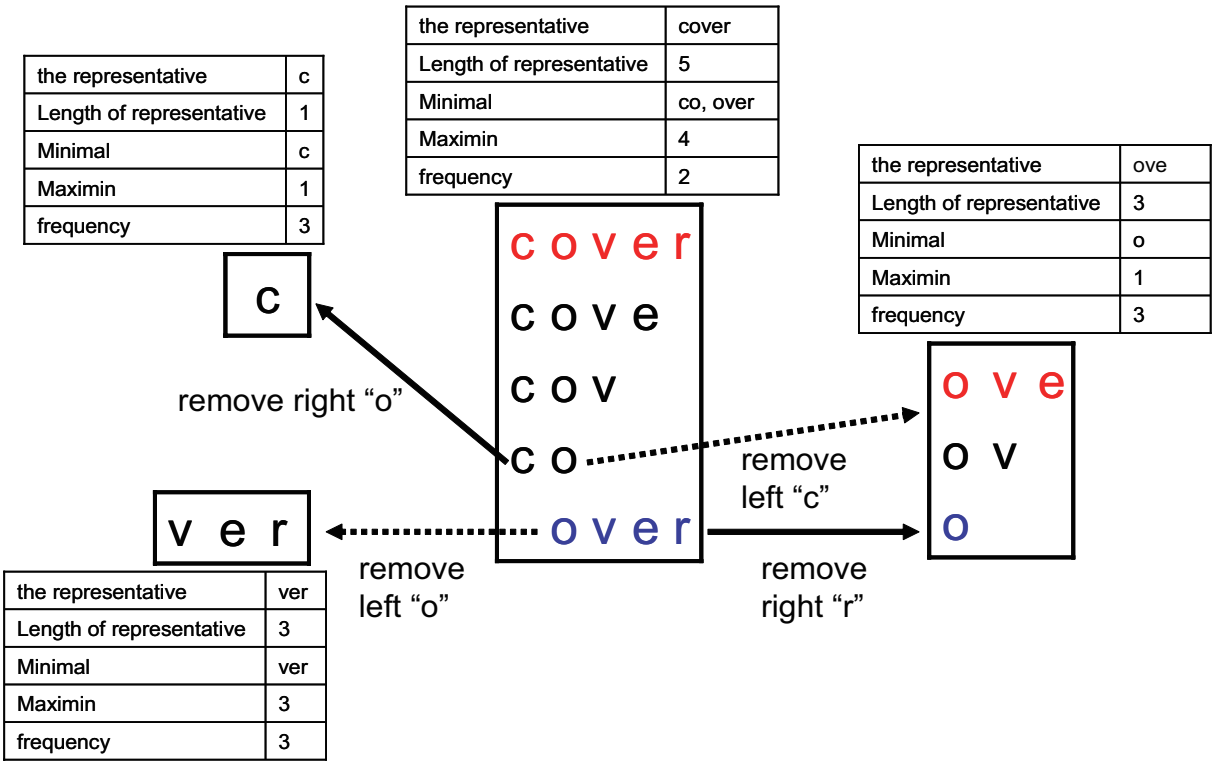

Figure 3: The example for $S = \{\texttt{discover}, \texttt{cover}, \texttt{November}, \texttt{vertical}\}$. The string "cover" is the representative of the equivalence class $[\texttt{cover}]_{\equiv} = \{\texttt{cover}, \texttt{cove}, \texttt{cov}, \texttt{co}, \texttt{over}\}$. $Minimal(\texttt{cover}) = \{\texttt{co}, \texttt{over}\}$, and $Maximin(\texttt{cover}) = |\texttt{over}| = 4$.

For any x in $Sub(S)$, let

$$BegPos_S(x) = \left\{ \langle w, j\rangle \;\middle|\; \begin{array}{l} w \in S, 0 \le j \le |w|, \\ x = w[j+1 : j+|x|] \end{array} \right\},$$
$$EndPos_S(x) = \left\{ \langle w, j\rangle \;\middle|\; \begin{array}{l} w \in S, 0 \le j \le |w|, \\ x = w[j-|x|+1, j] \end{array} \right\}.$$

For any $x \notin Sub(S)$, let $BegPos_S(x) = EndPos_S(x) = \emptyset$. In this paper, we omit the set S, and write simply $BegPos$ and $EndPos$.

For example, if $S = \{\texttt{discover}, \texttt{cover}, \texttt{November}, \texttt{vertical}\}$, then the sets $BegPos$ and $EndPos$ for their substrings are as follows.

$$\begin{aligned}
BegPos(\texttt{o}) &= BegPos(\texttt{ov}) = BegPos(\texttt{ove}) \\
&= \{\langle \texttt{discover}, 4\rangle, \langle \texttt{cover}, 1\rangle, \langle \texttt{November}, 1\rangle\}, \\
BegPos(\texttt{c}) &= \{\langle \texttt{discover}, 3\rangle, \langle \texttt{cover}, 0\rangle, \langle \texttt{vertical}, 5\rangle\}, \\
BegPos(\texttt{co}) &= BegPos(\texttt{co}) = BegPos(\texttt{cov}) \\
&= BegPos(\texttt{cove}) = BegPos(\texttt{cover}) \\
&= \{\langle \texttt{discover}, 3\rangle, \langle \texttt{cover}, 0\rangle\}, \text{and} \\
EndPos(\texttt{r}) &= EndPos(\texttt{er}) \\
&= \left\{ \begin{array}{l} \langle \texttt{discover}, 8\rangle, \langle \texttt{cover}, 5\rangle, \\ \langle \texttt{November}, 8\rangle, \langle \texttt{vertical}, 3\rangle \end{array} \right\}, \\
EndPos(\texttt{o}) &= \{\langle \texttt{discover}, 5\rangle, \langle \texttt{cover}, 2\rangle, \langle \texttt{November}, 2\rangle\}, \\
EndPos(\texttt{over}) &= EndPos(\texttt{cover}) \\
&= \{\langle \texttt{discover}, 5\rangle, \langle \texttt{cover}, 2\rangle\}.
\end{aligned}$$

Definition 2. Let x and y be arbitrary strings in Σ^*. The equivalence relations $\equiv_L$ and $\equiv_R$ are defined by

$$\begin{aligned}
x \equiv_L y &\Leftrightarrow BegPos(x) = BegPos(y), \\
x \equiv_R y &\Leftrightarrow EndPos(x) = EndPos(y).
\end{aligned}$$

The equivalence class of a string x in Σ^* with respect to $\equiv_L$ and $\equiv_R$ is denoted by $[x]_{\equiv_L}$ and $[x]_{\equiv_R}$, respectively.

For example, if $S = \{\texttt{discover}, \texttt{cover}, \texttt{November}, \texttt{vertical}\}$, then $[\varepsilon]_{\equiv_L} = [\varepsilon]_{\equiv_R} = \{\varepsilon\}$, $[\texttt{o}]_{\equiv_L} = [\texttt{ov}]_{\equiv_L} = [\texttt{ove}]_{\equiv_L} = \{\texttt{o}, \texttt{ov}, \texttt{ove}\}$, $[\texttt{c}]_{\equiv_L} = \{\texttt{c}\}$, $[\texttt{co}]_{\equiv_L} = [\texttt{cov}]_{\equiv_L} = [\texttt{cove}]_{\equiv_L} = [\texttt{cover}]_{\equiv_L} = \{\texttt{co}, \texttt{cov}, \texttt{cove}, \texttt{cover}\}$, and $[\texttt{r}]_{\equiv_R} = [\texttt{er}]_{\equiv_R} = \{\texttt{r}, \texttt{er}\}$, $[\texttt{o}]_{\equiv_R} = \{\texttt{o}\}$, $[\texttt{over}]_{\equiv_R} = [\texttt{cover}]_{\equiv_R} = \{\texttt{over}, \texttt{cover}\}$.

Definition 3. For any string x in $Sub(S)$, let $\overrightarrow{x}$ and $\overleftarrow{x}$ denote the unique longest members of $[x]_{\equiv_L}$ and $[x]_{\equiv_R}$, respectively.

For example, if $S = \{\texttt{discover}, \texttt{cover}, \texttt{November}, \texttt{vertical}\}$, then $\overrightarrow{\varepsilon} = \overleftarrow{\varepsilon} = \varepsilon$, $\overrightarrow{\texttt{o}} = \overrightarrow{\texttt{ov}} = \overrightarrow{\texttt{ove}} = \texttt{ove}$, $\overrightarrow{\texttt{c}} = \texttt{c}$, $\overrightarrow{\texttt{co}} = \overrightarrow{\texttt{cov}} = \overrightarrow{\texttt{cove}} = \overrightarrow{\texttt{cover}} = \texttt{cover}$, and $\overleftarrow{\texttt{r}} = \overleftarrow{\texttt{er}} = \texttt{er}$, $\overleftarrow{\texttt{o}} = \texttt{o}$, $\overleftarrow{\texttt{over}} = \overleftarrow{\texttt{cover}} = \texttt{cover}$.

Definition 4. For any string x in $Sub(S)$, let $\overleftrightarrow{x}$ be the string $\alpha x \beta$ such that α and β are the strings satisfying $\overrightarrow{x} = \alpha x$ and $\overleftarrow{x} = x\beta$.

For example, if $S = \{\texttt{discover}, \texttt{cover}, \texttt{November}, \texttt{vertical}\}$, then $\overleftrightarrow{\varepsilon} = \varepsilon$, $\overleftrightarrow{\texttt{o}} = \overleftrightarrow{\texttt{ov}} = \overleftrightarrow{\texttt{ove}} = \texttt{ove}$, $\overleftrightarrow{\texttt{c}} = \texttt{c}$, $\overleftrightarrow{\texttt{r}} = \overleftrightarrow{\texttt{er}} = \texttt{er}$, and $\overleftrightarrow{\texttt{co}} = \overleftrightarrow{\texttt{cov}} = \overleftrightarrow{\texttt{cove}} = \overleftrightarrow{\texttt{over}} = \overleftrightarrow{\texttt{cover}} = \texttt{cover}$.

Intuitively, $\overleftrightarrow{x} = \alpha x \beta$ means that:

- Every time x occurs in S, it is preceded by α and followed by β.
- Strings α and β are as long as possible.

Definition 5. Strings x and y are said to be *equivalent* on S if and only if:

1. $x \notin Sub(S)$ and $y \notin Sub(S)$, or
2. $x, y \in Sub(S)$ and $\overleftrightarrow{x}=\overleftrightarrow{y}$.

This equivalence relation is denoted by $\equiv$. The equivalence class of a string x in $Sub(S)$ with respect to $\equiv$ is denoted by $[x]_{\equiv}$.

For example, if $S = \{\texttt{discover}, \texttt{cover}, \texttt{November}, \texttt{vertical}\}$, then the strings in $Sub(S)$ are divided into the equivalence classes: $\{\varepsilon\}$, {o, ov, ove}, {c}, {r, er}, and {co, cov, cove, over, cover}.

A string x in $Sub(S)$ is said to be *prime* if $\overleftrightarrow{x}= x$. Let $Prime(S)$ denote the set of prime substring of S, that is,

$$Prime(S) = \{\overleftrightarrow{x} \mid x \in Sub(S)\}.$$

For example, if $S = \{\texttt{discover}, \texttt{cover}, \texttt{November}, \texttt{vertical}\}$, then $Prime(S) = \{$c, i, co, er, ve, ove, ver, cover, discover, November, vertical$\}$.

We regard each prime string x as the *representative* of the equivalence classes $[x]_{\equiv}$.

3.1.2 Properties on equivalence relation

For any x, y in Σ^*, we write $x \preceq y$ if x is a substring of y. For any x in $Prime(S)$, let $Minimal(x)$ denote the set of minimal elements of $[x]_{\equiv}$, that is,

$$Minimal(x) = \{y \in [x]_{\equiv} \mid \ z \preceq y \text{ and } z \in [x]_{\equiv} \text{ imply } z = y\}.$$

Let $Maximin(x)$ denote the maximum length of strings in $Minimal(x)$.

For example, if $S = \{\texttt{discover}, \texttt{cover}, \texttt{November}, \texttt{vertical}\}$, then $Minimal(\texttt{cover}) = \{\texttt{co}, \texttt{over}\}$ and $Maximin(x) = |\texttt{over}| = 4$.

Lemma 1. ([13]) For any x in $Prime(S)$, let $y_1, \ldots, y_k$ be the elements of $Minimal(x)$. Then,

$$[x]_{\equiv} = Pincer(y_1, x) \cup \cdots \cup Pincer(y_k, x),$$

where $Pincer(y_i, x)$ is the set of strings z with $y \preceq z \preceq x$.

3.2 Measures with Equivalence Classes

In this subsection, we give three measures for quantifying the alienness of equivalence classes. We use a data set obtained from the Web in order to evaluate these measures. The data set consists of postings from the YahooJapanFinance[1] forum. This forum is surveyed by the forum administrator, and postings are manually deleted if they are judged to be spam. Therefore, we can obtain spam and non-spam document examples by gathering the postings of a given forum over a certain period of time. We regard the postings which have been deleted as spams, and the writings not deleted as nonspams. This data set contains 1087 postings including 226 spam posts and 861 nonspam posts.

We consider the equivalence classes which appear at least twice in the documents, and do not regard strings only occurring once as spam.

[1] `http://quote.yahoo.co.jp/`

3.2.1 Length

In general, spams are different from natural sentences in that they tend to be lengthy and appear more frequently. Hence, we first consider the length of the representative of an equivalence class as a measure for spam detection:

$$measure_{Length}(x) = |\ \overleftrightarrow{x}\ |$$

There seems to exist a power law between the length of the representative of an equivalence class and the number of equivalence classes with the length (see Figure 4). In this plot, the blue asterisk point, denotes that all equivalence classes with that Length measure are substrings of spam documents only. The green cross point, denotes that all equivalence classes with that Length measure are substrings of nonspam documents only. The red plus point denotes otherwise, and equivalence classes with that Length measure are included in both spam and nonspam documents.

In this plot, we can see that spam equivalence classes are distributed on high length parts. We can say that an equivalence class has a high degree of probability for being spam if the length of the representative of the equivalence class is long.

Figure 7 shows the ROC curve for the Length measure. The x-axis is the negative ratio for each equivalence class, that is

$$negative\ ratio = \frac{\#\ of\ nonspam\ documents}{\#\ of\ detected\ documents},$$

and y-axis is the positive ratio for each equivalence class, that is

$$positive\ ratio = \frac{\#\ of\ spam\ documents}{\#\ of\ detected\ documents}.$$

The graph is drawn by considering all possible Length measure values as a threshold, and plotting the above values by classifying documents which contain the equivalence class whose length is longer than the threshold value as spam, and nonspam otherwise. As can be seen in Figure 7, the length of the representative of an equivalence class seems to be an effective measure for spam detection.

3.2.2 Size

Next, we consider the size of an equivalence class as a measure for spam detection:

$$measure_{Size}(x) = |[x]_{\equiv}|$$

Although there is a strong relationship between the length of the representative of equivalence classes and the size of equivalence classes (see Figure 16), there are equivalence classes whose representative is long, but whose size is not large. There also seems to exist a power law between the size of an equivalence class and the number of equivalence classes with the size (see Figure 5). In this plot, there are more spam specific points than that of the length measure when the measurement is high.

Figure 8 is the ROC curve for the measure of the size. This curve show that the measure of the size is also effective.

3.2.3 Maximin

The Maximin measure is defined as the difference between the length of the representative and the length of the longest minimal element of the equivalence class:

$$measure_{Maximin}(x) = |\ \overleftrightarrow{x}\ | - Maximin(x)$$

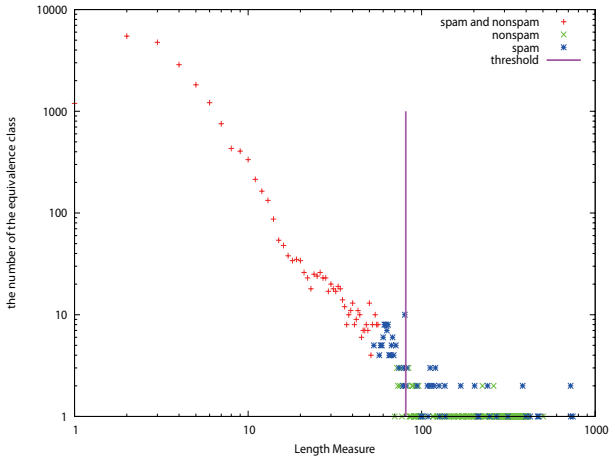

Figure 4: Length measure distribution of equivalence classes. The threshold line is obtained by our method described in Section 3.3.

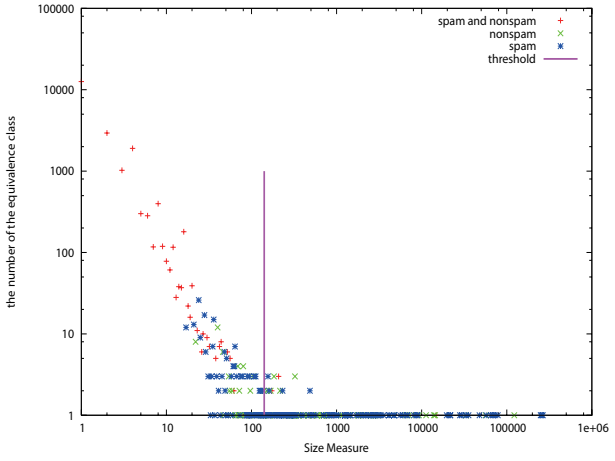

Figure 5: Size measure distribution of equivalence classes.The threshold line is obtained by our method described in Section 3.3.

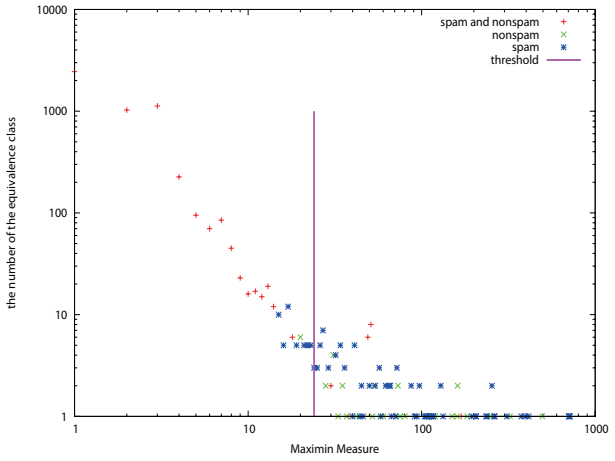

Figure 6: Maximin distribution of equivalence classes.The threshold line is obtained by our method described in Section 3.3.

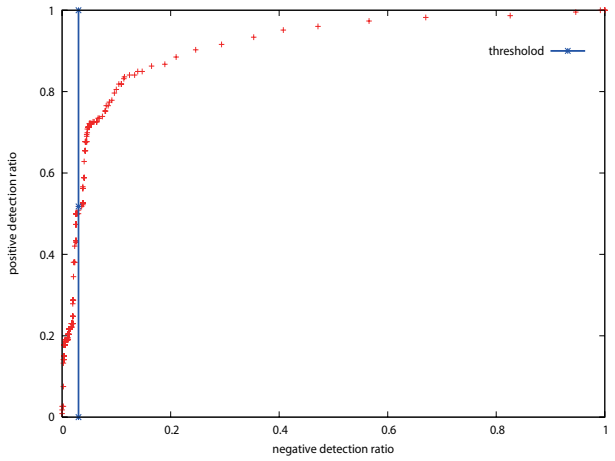

Figure 7: ROC curve of the length measure for discriminating between spam and nonspam documents.

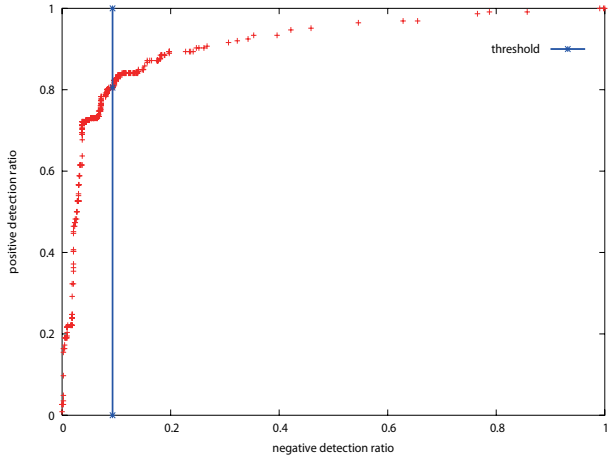

Figure 8: ROC curve of the size measure for discriminating between spam and nonspam documents.

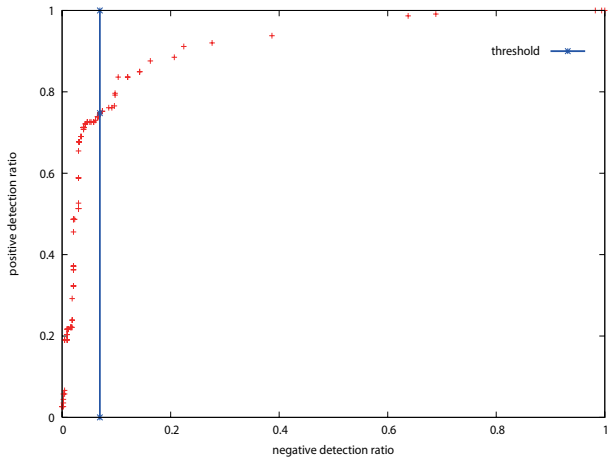

Figure 9: ROC curve of the Maximin measure for discriminating between spam and nonspam documents.

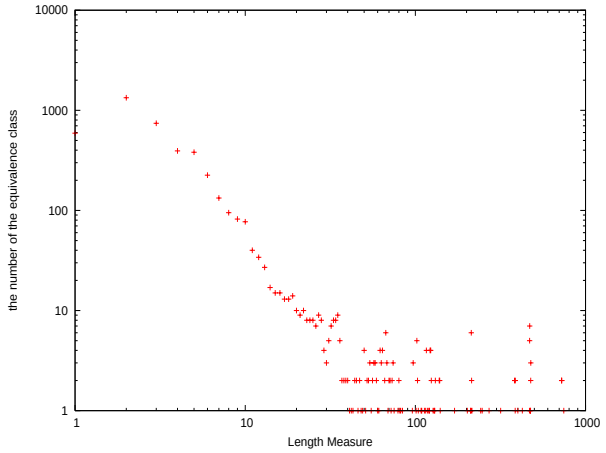

Figure 10: Length measure distribution of equivalence classes for spam documents.

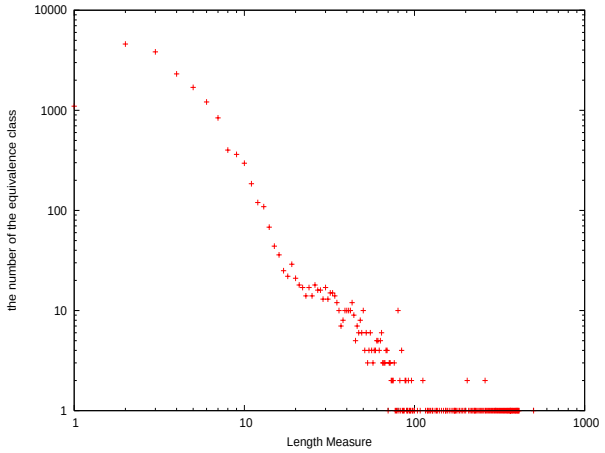

Figure 13: Length measure distribution of equivalence classes for only nonspam documents.

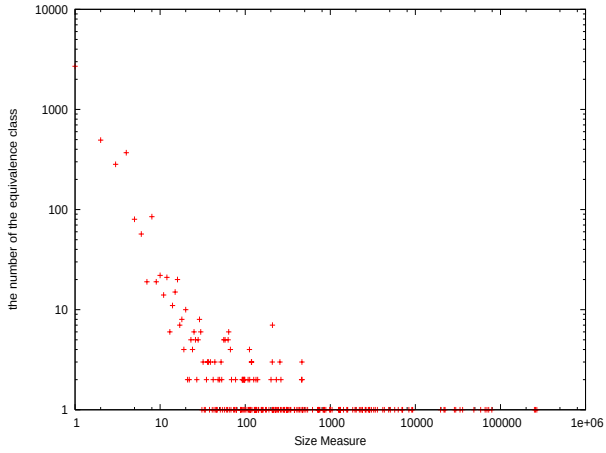

Figure 11: Size measure distribution of equivalence classes for only spam documents.

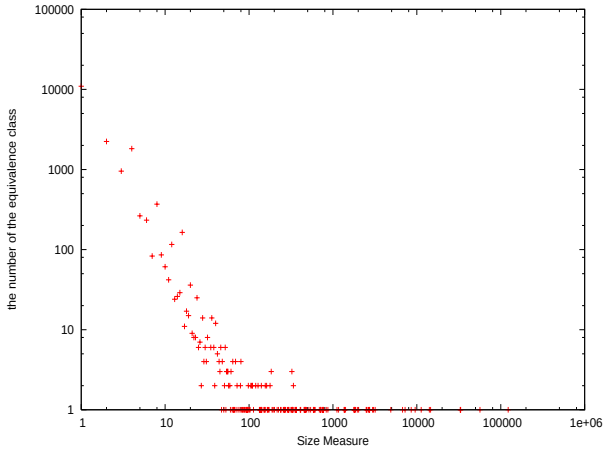

Figure 14: Size measure distribution of equivalence classes for only nonspam documents.

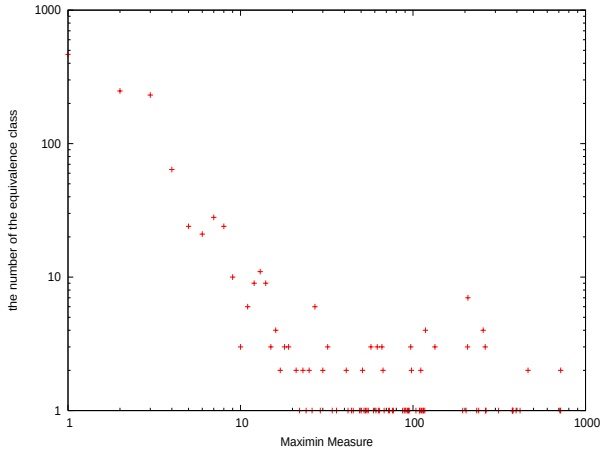

Figure 12: Maximin measure distribution of equivalence classes for only spam documents.

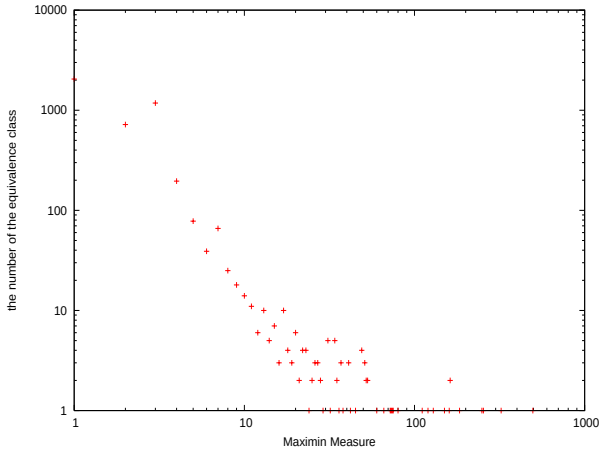

Figure 15: Maximin measure distribution of equivalence classes for only nonspam documents.

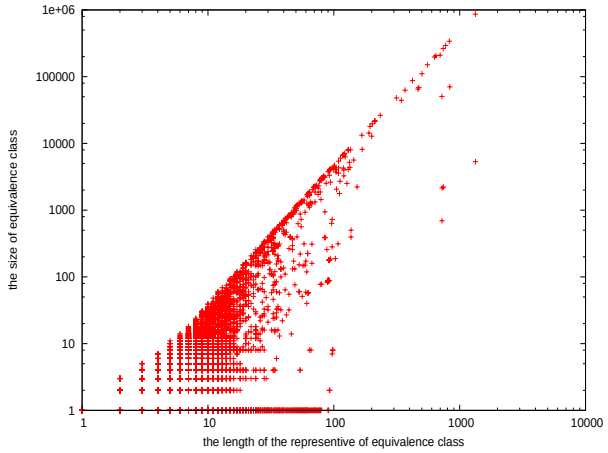

Figure 16: Relation between the length of the representative of the equivalence class and the size of the equivalence class.

Table 1: The area under the ROC curve for each measure.

negative ratio	≤ 0.3	≤ 0.5	≤ 1.0 (all)
Length	0.282	0.485	0.913
Size	0.243	0.469	0.915
Maximin	0.318	0.559	0.919

This measures the alienness of a substring equivalence class as the distance from the longest equivalence class that is contained in it. Figure 6 shows the relation between the $measure_{Maximin}(x)$ and the number of the equivalence classes with the measure.

Figure 9 shows the ROC curve for this measure, showing that this measure is also effective for spam detection. In addition, the area under the ROC curve given the negative ratio is less than 0.3 is larger than those of other measures (see Table 1). Hence, we expect that this measure has lower false-positive error.

3.3 Determining the Threshold Value

Our method takes the unlabeled document set as input. We use a simple threshold to determine whether or not a given measure value is "alien" or not. Documents are then judged as spam if it includes an "alien" substring equivalence class. Below, we describe a simple unsupervised method for determining the threshold value.

Looking at each of the measure plots, Figure 4, Figure 5 and Figure 6, we can see that there are roughly two parts. One is the spam specific part on right area, and the rest is on the left area. We propose heuristics to find a point that separates the spam part from the nonspam part, and regard the point as the threshold. More precisely, we model each of the two parts using a linear model, and we look for the point of separation where the two linear models best explains the data points.

Let $S = ((x_1, y_1), \ldots, (x_n, y_n))$ be a sequence of n points, where $x_1 \leq x_2 \leq \cdots \leq x_n$. For $1 \leq k < n$, let S_1^k be the sequence of the first k points in S, and let S_2^k be the remaining sequence in S. We choose the k^*-th point that minimizes the sum of the least square errors in the left and the right sides of points. That is, we choose

$$k^* = \arg \min_{1 \leq k < n} (LSE(S_1^k) + LSE(S_2^k)),$$

where

$$LSE(S') = \min_{a,b} \sum_{i=1,\ldots,n'} (y'_i - ax'_i - b)^2$$

for $S' = ((x'_1, y'_1), \ldots, (x'_{n'}, y'_{n'}))$. It is well known that the solution of $LSE(S')$ is obtained if

$$a = \frac{n' \sum_{i=1}^{n'} x'_i y'_i - \sum_{i=1}^{n'} x_i \sum_{i=1}^{n'} y'_i}{n' \sum_{i=1}^{n'} {x'_i}^2 - (\sum_{i=1}^{n'} x'_i)^2},$$

and

$$b = \frac{n \sum_{i=1}^{n'} {x'_i}^2 \sum_{i=1}^{n'} y'_i - \sum_{i=1}^{n'} x'_i y'_i \sum_{i=1}^{n'} x'_i}{n' \sum_{i=1}^{n'} {x'_i}^2 - (\sum_{i=1}^{n'} x'_i)^2}.$$

4. COMPUTATIONAL EXPERIMENTS

We detect spams in four forums of Yahoo Japan Finance [2], which are collected in the same way as the data used for evaluating the measures in Section 3. If spam is posted in these forums, the spam is manually deleted by the forum administrator. We regard the deleted posts as spam.

We selected and collected data from four forums: 4314[3], 4974[4], 6830[5], 8473[6]. We detect spams from these four data sets using the three measures proposed in Section 3. The results are shown in Table 2, where the values Recall, Precision, F-score in the table are defined as follows:

$$Recall = \frac{\#\ of\ detected\ spam\ documents}{\#\ of\ spam\ documents}$$

$$Precision = \frac{\#\ of\ detected\ spam\ documents}{\#\ of\ detected\ documents}$$

$$F\text{-}score = \frac{2 * Recall * Precision}{Recall + Precision}$$

As shown in Table 2, the measure Maximin has the highest F-scores, which vary from 68% to 80%, among three measures for all of the data sets. On the other hand, as for recall values, the measure Size outperforms others, while its F-scores remain close to those of Maximin. We also evaluate the three measures when the inputs are nonspam documents only. The percentages of documents judged as spams in this setting are summarized in the column "nonspam" of Table 2. Note that the value nonspam should be 0% ideally. The measure Length has the lowest nonspam values over all the data sets. In summary, none of the three measures completely outperforms the others.

We examined the false positive strings (nonspams which our method judges to be spams) and the false negative strings (spams which our method judges to be nonspams). In the former case, most of false positive strings are difficult to distinguish from spams by their contents, even for human. In the latter case, false negative strings contain many abusive languages, which are not spams in general but are deleted

[2] http://quote.yahoo.co.jp
[3] http://messages.yahoo.co.jp/?action=q&board=4314
[4] http://messages.yahoo.co.jp/?action=q&board=4974
[5] http://messages.yahoo.co.jp/?action=q&board=6830
[6] http://messages.yahoo.co.jp/?action=q&board=8473

Table 2: The results of each measure for YahooJapanFinance forum data.

forum	total data size	# of spam	# of nonspam	measure	Recall (%)	Precision (%)	F-score (%)	time(sec) Measure	time(sec) threshold	time(sec) total	nonspam (%)
4314	484 KB	291	1424	Length	45	57	50	0.60	0.02	0.62	**2.67**
				Size	**89**	60	72	0.57	0.75	1.34	7.51
				Maximin	80	**80**	**80**	**0.58**	**0.00**	**0.58**	6.18
4974	552 KB	331	1315	Length	39	**77**	52	0.98	0.03	1.01	**4.33**
				Size	**63**	69	66	0.96	7.30	8.26	11.56
				Maximin	60	75	**67**	**0.94**	**0.02**	**0.96**	9.81
6830	588 KB	317	1613	Length	40	**72**	52	0.90	0.08	0.98	**3.47**
				Size	**74**	57	64	0.87	1.63	2.50	17.73
				Maximin	69	69	**69**	**0.86**	**0.01**	**0.87**	13.95
8473	620 KB	264	1597	Length	57	**76**	65	0.79	0.02	0.81	**4.32**
				Size	**72**	63	67	**0.77**	3.40	4.27	8.14
				Maximin	67	69	**68**	**0.77**	**0.01**	**0.78**	9.39

by the forum administrator. So, in a practical point of view, our method detects spams well.

The time required for detecting spam using the Size measure is longer than the other two measures. We note that the time required for enumerating the equivalence classes and calculating the measures of each equivalence class does not vary for the three measures. However, since the Size measure can have a wider range of values, it will in general have more points on the distribution plot (see Figure 5 to Figure4 and Figure 6). This affects the time for calculating the linear regression, and finding the threshold value for discriminating spam and nonspam documents.

5. CONCLUSIONS

We proposed a new, simple, unsupervised method for spam detection based on the alienness of strings. We proposed three such measures, namely Length, Size and Maximin of the string equivalence classes. We observed that spam documents seem to give rise to equivalence classes with measurements larger than nonspam documents, and developed a method for finding a threshold value for discriminating between spam and nonspam equivalence classes. The results on actual data obtained from web forum postings shows that our methods detects spams fairly well. We expect that more accurate spam detection would be possible by analyzing and exploiting the properties of the distributions induced by equivalence classes.

6. REFERENCES

[1] L. Becchetti, C. Castillo, D. Donato, S. Leonardi, and R. Baeza-Yates. Link-based characterization and detection of web spam. In *AIRWEB'06*, 2006.

[2] A. A. Benczú, K. Csalogány, and T. Sarlós. Link-based similarity search to fight web spam. In *AIRWEB'06*, 2006.

[3] A. Blumer, J. Blumer, D. Haussler, R. Mcconnell, and A. Ehrenfeucht. Complete inverted files for efficient text retrieval and analysis. *J. ACM 34(3)*, pages 578–595, 1987.

[4] CNETNEWS.COM. Spim, splog on the rise, November 2006. `http://news.com.com/2100-7349_3-6091123.html`.

[5] A. L. da Costa Carvalho, P. A. Chirita, E. S. de Moura, P. Calado, and W. Nejdl. Site level noise removal for search engines. In *World Wide Web Conference*, 2006.

[6] B. J. Jansen. Adversarial information retrieval aspects of sponsored search. In *AIRWEB'06*, 2006.

[7] T. Kasai, G. Lee, H. Arimura, S. Arikawa, and K. Park. Linear-time Longest-Common-Prefix Computation in Suffix Arrays and Its Applications. In *Proceeding of the 12th Annual Symposium on Combinatorial Pattern Matching*, Lecture Notes in Computer Science 2089, pages 181–192. Springer-Verlag, 2001.

[8] U. Manber and G. Myers. Suffix arrays: a new method for on-line string searches. *SIAM Journal on Computing*, 22(5):935–948, 1993.

[9] K. Narisawa, H. Bannai, S. Inenaga, and M. Takeda. Space-economical computation of substring equivalence classes. Unpublised manuscript, 2006.

[10] K. Narisawa, Y. Yamada, D. Ikeda, and M. Takeda. Detecting blog spams using the vocabulary size of all substrings in their copies. In *Proceedings of the 3rd Annual Workshop on Weblogging Ecosystem*, 2006.

[11] A. Ntoulas, M. Najork, M. Manasse, and D. Fetterly. Detecting spam web pages through content analysis. In *World Wide Web Conference*, 2006.

[12] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz. A Bayesian Approach to Filtering Junk E-mail. *AAAI Workshop on Learning for Text Categorization*, July 1998. `ftp://ftp.research.microsoft.com/pub/ejh/junkfilter.pdf`.

[13] M. Takeda, T. Matsumoto, T. Fukuda, and I. Nanri. Discovering characteristic expressions in literary works. *Theoretical Computer Science*, 2003.

[14] B. Wu, V. Goel, and B. D. Davison. Topical trustrank: Using topicality to combat web spam. In *World Wide Web Conference*, 2006.

[15] K. Yoshida, F. Adachi, T. Washio, H. Motoda, T. Homma, A. Nakashima, H. Fujikawa, and K. Yamazaki. Density-based spam detector. In *KDD'04*, pages 486–493, 2004.

[16] G. K. Zipf. *The Psycho-Biology of Language: An Introduction to Dynamic Philology*. Houghton Mifflin, 1935.

[17] G. K. Zipf. *Human Behavior and the Principle of*

Least Effort. Addison-Wesley, 1949.