Hindawi
Security and Communication Networks
Volume 2019, Article ID 6587020, 16 pages
https://doi.org/10.1155/2019/6587020

Research Article

Practical Web Spam Lifelong Machine Learning System with Automatic Adjustment to Current Lifecycle Phase

Marcin Luckner

Faculty of Mathematics and Information Science, Warsaw University of Technology, Koszykowa 75 Street, 00-662 Warsaw, Poland

Correspondence should be addressed to Marcin Luckner; mluckner@mini.pw.edu.pl

Received 25 September 2018; Revised 14 January 2019; Accepted 29 January 2019; Published 20 February 2019

Guest Editor: Rafal Kozik

Copyright © 2019 Marcin Luckner. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Machine learning techniques are a standard approach in spam detection. Their quality depends on the quality of the learning set, and when the set is out of date, the quality of classification falls rapidly. The most popular public web spam dataset that can be used to train a spam detector—WEBSPAM-UK2007—is over ten years old. Therefore, there is a place for a lifelong machine learning system that can replace the detectors based on a static learning set. In this paper, we propose a novel web spam recognition system. The system automatically rebuilds the learning set to avoid classification based on outdated data. Using a built-in automatic selection of the active classifier the system very quickly attains productive accuracy despite a limited learning set. Moreover, the system automatically rebuilds the learning set using external data from spam traps and popular web services. A test on real data from Quora, Reddit, and Stack Overflow proved the high recognition quality. Both the obtained average accuracy and the F-measure were 0.98 and 0.96 for semiautomatic and full–automatic mode, respectively.

1. Introduction

Despite several existing algorithms for web spam detection, which were, for example, presented in works [1–11], unsolicited messages are still a major issue for the Internet community. The main reason is the evolution of web spam. Several works have shown that the classifiers learned from old data cannot recognise new web spam cases [1, 11–14]. Several solutions have been proposed to avoid the issue of outdated learning sets [1, 2, 11, 15, 16], but the obtained results have only been a partial success. The reduction of quality over time was limited but not stopped. Moreover, most of the solutions ignored the lack of appropriate learning sets during initiation of a web spam recognition system in a new site.

In practice, web spam recognition should include a lifelong learning mechanism. The system—dedicated to protecting a specific site—should evolve during its lifecycle according to the current phase identified by knowledge of spam and nonspam data.

Let us discuss the lifecycle of a web spam recognition system presented in Figure 1. At first, the system is initialised to protect a new site (a new blog or forum). The history of previously published comments or entries does not exist. Therefore, during the initial starting-up phase, the classifier cannot learn from patterns of typical nonspam messages. The classification of all messages is based on spam examples only. The spam examples—unlike the nonspam ones—have a global character and can be collected using a honey pot.

The next phase is the development phase. In this phase, a set of regular nonspam data already exists, but the number of entries is relatively small in comparison to known spam examples. An appropriate classifier to analyse the entries is a method developed for imbalanced sets. Other methods tend to classify all the entries to the dominating class.

When a sufficient number of nonspam messages are collected, the system enters the mature phase. During this third phase, the number of nonspam messages is growing. Now, the system can use the messages to create a balanced dataset from both types of messages. The classification during this phase brings the best results.

Over time, the system enters the descending phase. New spam messages are significantly different from the ones used for the initialisation of the system. As a result, the classification accuracy fails regularly.

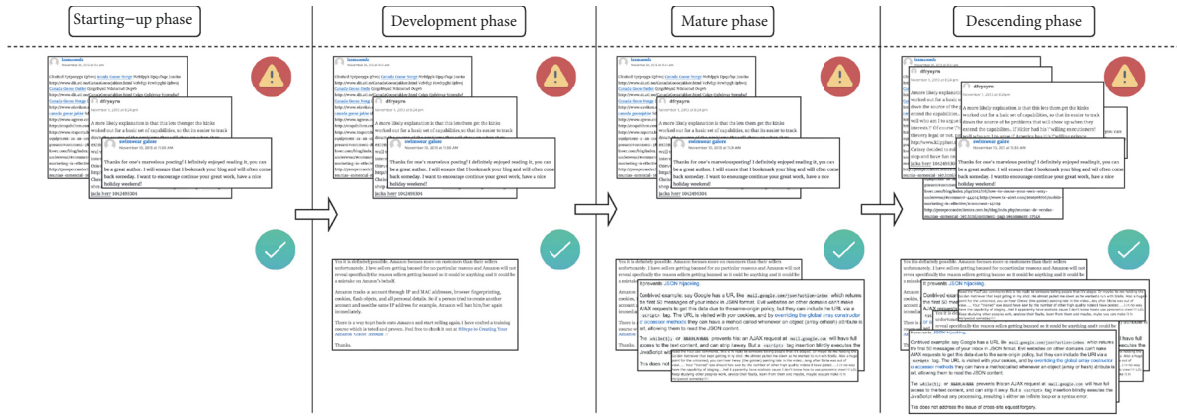


Figure 1: Lifecycle of web spam recognition system according to known spam and nonspam data.

Most of the works on spam detection propose a classification algorithm that is based on a stable learning set and masters the accuracy obtained from a static learning set. The evolution of the classification process is not discussed, or it is limited to a specific issue as a reduction of the recognition accuracy over time.

In this work, we have proposed a new system for spam classification that addresses all problems connected with the described phases. The novelty of our proposition lies in an automaton that selects a proper classification methodology according to the current phase of the system lifecycle. Additionally, the proposed system offers full automatisation in the creation of the learning sets using an external data source. Therefore, the system can be implemented by raising blog platforms without substantial datasets of labelled comments.

We focused on web spam as it is still one of the most challenging issues. The most common type in the list is web spam that exploits vulnerabilities and gaps in the web 2.0 to inject links to spam content into dynamic and shareable content such as blogs, comments, reviews, or wiki pages.

We tested the system on three popular web services: Quora, Reddit, and Stack Overflow. The test proved high recognition quality. Using the datasets, we discussed practical issues of lifelong machine learning, including the spam classification in the case of an insufficient number of nonspam learning examples and the descending accuracy of the system over time.

The remainder of this work is organised as follows. First, other works on web spam detection are described in Section 2. That is followed by a description of the proposed dynamic web spam recognition system in Section 3. Section 4 presents the testing methodology. Section 5 contains the results, while Section 6 discusses the obtained results. That is followed by the conclusions in Section 7.

2. Related Work

Several works have analysed the web spam recognition issue. In works [23–25], the authors summarised and compared the results obtained by various machine learning techniques used for web spam recognition.

Some original approaches were proposed in the following works. Yin et al. [26] proposed a general mathematical framework, which proves beneficial for grouping classifiers using a convex ensemble diversity measure. Goh et al. [10] incorporated weight properties to enhance the web spam detection algorithms. The weight properties were defined as the influences of one Web node on another one. Manaskasemsak et al. [27] discussed the ant colony optimisation designed to let an ant start from a nonspam seed and a compilation of the created path to nonspam classification rules.

The problem of the decrease in classification accuracy over time was stressed in [1, 2, 11, 15, 16]. The works showed—using WEBSPAM-UK2006 and WEBSPAM-UK2007 datasets [28]—that it is impossible to keep the same classification quality over a number of years. We present a new approach to this issue. Our spam rejector evolves over time. Some other works proposed a dynamically changing spam detector before. Hao et al. [29] presented a method for detecting domains used for email spamming at the time of registration and evaluated their method with sliding time windows for training and testing data. Veeramachaneni et al. [30] described a system that combined outlier detection methods with a supervised learning module to improve the accuracy of intrusion detection. The supervised learning module is parametrised with the learning time window. We replaced the time factor by the size of the learning set. The two approaches are compared in Section 6.3.

Our solution depends on features that discriminate against web spam. Several works proposed their own set of features. Alarifi et al. [31] examined changes in the distribution of the set of selected detection features according to the page language. Dai et al. [17] proved that historical web page information was an essential factor in spam classification. Works [32, 33] showed the importance of the analysis of URL addresses in spam detection. Urvoy et al. [34] described how to use the HTML style similarity cluster to pinpoint dubious pages and enhance the quality of spam classifiers. Piskorski et al. [35] proved that specific linguistic features could be useful for a spam-detection task when combined with features studied elsewhere. Also, Islam et al. [36] used

tokens (words) to create a set of 21 hand-crafted features from each e-mail message to recognise spam. The abovementioned papers inspired part of the features used in our system. We used a data mining process to select useful features among the created ones. A similar approach is presented in several works, i.e., [4, 37].

Our rejector is based on Deterministic Finite Automaton. Dolzhenko et al. [38] also proposed a model called mandatory results automata (MRAs). MRAs could monitor and transform security-relevant actions and their results. However, the model was theoretical and not tested on real data. Another interesting application was presented in [39]. The authors proposed estimator learning automaton-based subset selection as a new method for feature selection in high-dimensional spaces.

We propose a complete system for web spam rejection from blogs. The following systems aimed at email spam were created before. Bruckner et al. [16] evaluated spam filters derived from different optimisation problems on chronologically ordered emails. The Nash–equilibrial prediction models used outperformed reference methods. However, the execution time is 10 thousand times higher for the Nash-equilibrial prediction models than for Support Vector Machines (SVM). Colbaugh et al. [15] presented two mechanisms. The predictive defence algorithm combined game theory with machine learning to model and predict future adversary actions for synthesising robust defences. The extrapolative defence algorithm involves extrapolating the evolution of defence configurations forward in time, as a way of generating defences. A new web spam filtering framework (WSF2) was presented in works [40, 41]. The system proposed by the authors dynamically adjusts different parameters to ensure continuous improvement in filtering precision over time. The framework was tested using combinations of different filtering techniques including regular expressions and well-known classifiers.

3. Methodology

The main feature of the proposed system is the dynamic selection of a classification method according to the current phase of the system lifecycle. The lifecycle is modelled by a finite automaton that switches spam rejectors according to data flow.

A second important aspect is the collection of learning data from third-party sources to keep the system up to date even if the flow of data is too small to fulfill machine learning requirements.

3.1. Web Spam Rejection. Let us assume that set S^- contains comments on a web platform. The comments are described by a feature vector $\mathbf{x} \in \mathbb{R}^m$ and belong to n classes, where $n \geq 1$. The classes can describe topics, usefulness, aggression levels, or other. The classes are detected by the classification function $f : \mathbb{R}^m \longrightarrow 1 \dots n$.

Let us define set S^+ as the set of web spam comments. Web spam comments are generated automatically. Mostly, such comments cannot be correctly classified to any of the known classes. Results of the classification function f on set S^+ are not useful and obscure evaluation of the classification function. Therefore, we need an independent method—not integrated with f function—to eliminate web spam comments.

Let us define a membership function for elements of the known classes $\mu : \mathbb{R}^m \longrightarrow 0 \dots n$. Elements $\{\mathbf{x} \in \mathbb{R}^m : \mu(\mathbf{x}) > 0\}$ are elements of known classes called *native elements*. When the membership function returns zero the element does not belong to any of classes. Such elements are called *foreign elements*. The task of the rejection function is the separation of the *native elements* from the *foreign elements*. In our case, the rejection function eliminates spam from the comments.

For that, the rejection function is used on the elements from the set $S = S^+ \cup S^-$ to create a new set $\overline{S} = \{\mathbf{x} \in \mathbb{R}^m : R(\mathbf{x}) > 0\}$. The number of the *foreign elements* in the set should be reduced. The reduction task can formulate the following minimisation condition:

$$\min_{\overline{S}} \left(\frac{\sum_{\mathbf{x} \in \overline{S}} 1 - \rho\left(\mu\left(x\right)\right)}{|S|} \right), \tag{1}$$

where $\rho(x) = 1 \Longleftrightarrow x \neq 0$ and $\rho(0) = 0$.

Formula (1) reaches a minimum if $\overline{S} \subseteq S^+$. However, the condition is also fulfilled if $\overline{S} = \emptyset$ and that means the elimination of the *native elements*. Therefore, a second condition is introduced. The condition, in contrast to the previous one (1), is a maximisation condition

$$\max_{\overline{S}} \left(\frac{\sum_{\mathbf{x} \in \overline{S}} \mu\left(\mathbf{x}\right)}{\sum_{\mathbf{x} \in S} \mu\left(x\right)} \right). \tag{2}$$

If Formula (1) and Formula (2) obtain the minimum and maximum, respectively, then $\overline{S} = S^-$ and all spam examples are eliminated.

In practice, the rejection can be implemented as a binary classification of the native and foreign elements [42]. However, the quality of implementation depends on the knowledge of both classes. In machine learning techniques, the knowledge of classes comes from the learning set. Let us define two learning sets $L_0 \subset S^+$ and $L_1 \subset S^-$ as

$$L_i = \left\{\mathbf{x} \in \mathbb{R}^m : \rho\left(\mu\left(\mathbf{x}\right)\right) = i\right\} \wedge i = 0 \dots 1. \tag{3}$$

We can define three cases according to the learning sets. In the first case, the learning set is $L_1 = \emptyset$. That raises two problems. First, the group of one-class classifiers that we can implement is limited. Second, the quality of the classifier is doubtful because Formula (1) cannot be used in the evaluation process and the whole optimisation is limited to Formula (2).

In the second case, $L_1 \neq \emptyset$ but $L_1 << L_0$. That allows us to optimise both conditions. However, the disproportion between the learning sets may cause a bias in the classification process.

The third case assumes that the sizes of the learning sets are similar. This case is the most desirable. However, if the number of learning cases is too high, the classifier may not be flexible enough to learn new forms of spam and its quality may decrease in time.

A good web spam detection system should work correctly in all three cases. Therefore, we propose, for web spam rejection, a finite automaton, which switches between various types of rejections according to the parameters of the learning set.

3.2. Automaton. Let us define a finite automaton $M = (Q, \Sigma, \delta, q_0, F)$. A finite set of states $Q = \{R_0, R_1, R_2, I_0, I_1, I_2\}$ contains the start state $q_0 = R_0$ and the subset of final states $F = \{R_0, R_1, R_2\}$. A finite input alphabet contains symbols in the form r_i^s, where $s \in \{+, -\}$ and $i = 0 \dots 2$.

The symbols describe the following rules based on the cardinal number of the sets of nonspam S^- and spam S^+ messages:

r_0: $|S^-| \geq k$, the number of nonspam messages is high enough to create a binary classifier,

r_1: $|S^-|/|S^+| \geq l$, the number of nonspam messages is similar to the number of spam messages and a balanced classifier can be created,

r_2: $|S^-| \geq m$, the number of nonspam messages is too high to learn new spam forms.

The notation r_i^+ means that the i-th rule is fulfilled, while r_i^- means otherwise.

The transition function $\delta : Q \times \Sigma \longrightarrow Q$ is defined by the following transitions:

$$\begin{aligned}
\delta\left(R_i, r_i^+\right) &= R_{i+1} \wedge i = 0 \dots 1,\\
\delta\left(R_i, r_{i-1}^-\right) &= R_{i-1} \wedge i = 1 \dots 2,\\
\delta\left(R_i, r_i^-\right) &= I_i \wedge i = 0 \dots 2,\\
\delta\left(R_2, r_2^+\right) &= R_2,\\
\delta\left(I_i, r_i^+\right) &= R_{i+1} \wedge i = 0 \dots 1,\\
\delta\left(I_i, r_i^-\right) &= I_i \wedge i = 0 \dots 2,\\
\delta\left(I_2, r_2^+\right) &= R_2.
\end{aligned} \tag{4}$$

Figure 2 shows the structure of the automaton. While some final state is reached, the system rebuilds the web spam rejector using a new learning dataset. In states I_i, the system uses the lastly created classifier because there is no need to create a new one.

In most cases, a more extensive dataset allows us to create a better rejector and the automaton goes from R_0 to R_2 when new data is being collected. However, there are some exceptions. Firstly, the automaton can go back to the previous final state when changes in the sets break a relevant rule. This is possible if we have a rapid growth of spam messages or if some nonspam messages are removed from the system by a moderator. Secondly, when the learning set is too big and too old, the classifier must be rebuilt. This is done in states R_2 and I_2. The new classifier is trained on selected newest messages.

There is some similarity between our model and the model proposed in [30], which switches between outlier detection and supervised learning. However, there are several important differences between two models. Our model changes states using three rules r_0, r_1, and r_2 based on relations between classes in the learning set. Meanwhile, the model proposed in [30] switches between two states, where the initial unsupervised state is activated for a minimal set of events. Moreover, the dynamic learning set is based on a time factor when we analyse the structure of the set to rebuild it. We compare and discuss those two approaches in Section 6.3.

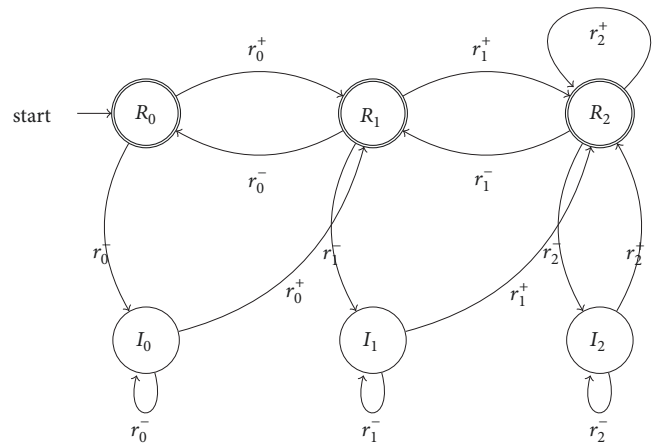


Figure 2: Automaton switches between spam rejectors according to the size of the known nonspam and spam sets.

3.3. System. Two aspects distinguish the proposed web spam detection solution from the others. The first aspect is the dynamic classification module that chooses a proper classification method according to the current phase of the system lifecycle. The automaton that models the lifecycle was described in Section 3.2. The second aspect is the mechanism of learning set creation that allows the operator to use external data sources to increase recognition quality especially in the initial phase of a protected web service.

The creation of the learning set from the labelled documents is critical for the quality of the system. Figure 3 presents a detailed schema of this process.

The learning set $L_0 \cup L_1$, used in the system, consists of two sets: the learning web spam set L_0 and the learning nonspam set L_1. The system can work even if one of the learning subsets is empty.

The learning set can be created from several independent sources defined as subsets of spam data S^+ and nonspam data S^- introduced in Section 3.1. We discuss two approaches to labelling the data.

The classification subsystem identifies the incoming comments on the protected web services. Usually, the classification is not perfect and must be supervised by the operator. As a result, two sets labelled internally as spam and nonspam are called inS^+ and inS^-, respectively.

The system uses external datasets. The first set exS^+ contains spam examples collected by fake WordPress blogs working as a honeypot. The second set exS^- consists of

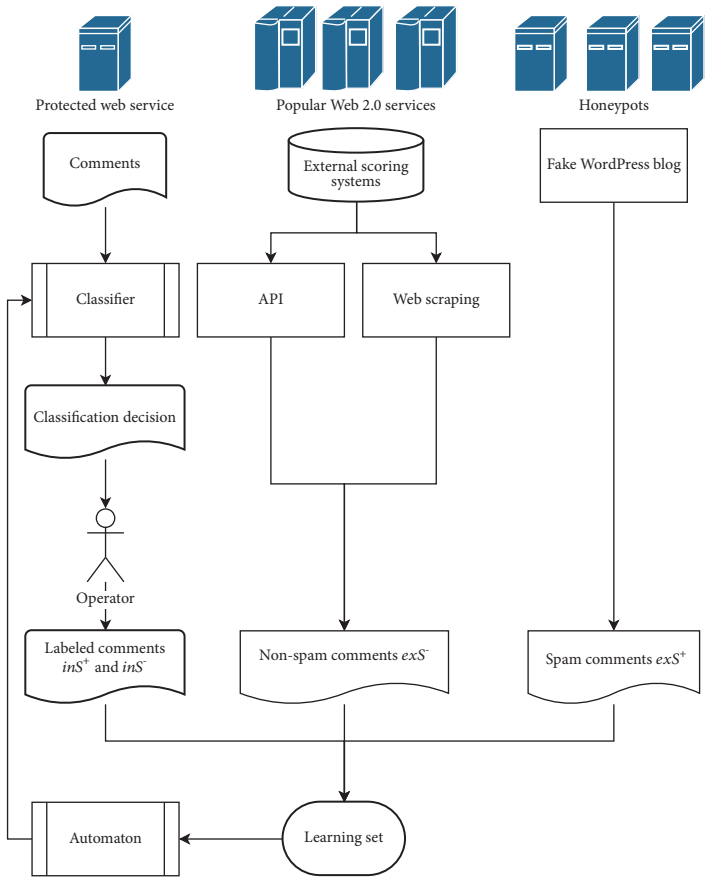


Figure 3: Schema of learning set creation system.

Table 1: Sets notation.

Type	Source	Notation
Spam	All	S^+
Spam	Internal	inS^+
Spam	External	exS^+
Non-spam	All	S^-
Non-spam	Internal	inS^-
Non-spam	External	exS^-

nonspam comments from various popular web 2.0 services with a scoring system. The notation for data is summarised in Table 1.

Our system communicates with the scoring system through an existing API or by web scraping to collect comments commonly acknowledged as nonspam. The technical details of the comments collection are presented in Section 4.3. The discussion of usage of external sets is presented in Section 6.2.

The automaton controls the classification in the system. The automaton reacts to changes in the learning set. For a small set, specialised machine learning algorithms are used to create an appropriate classifier. A proposition of implementation is given in Section 4.1.

When the set is too large, the learning set is relaxed. The relaxation process removes old data from the learning set and rebuilds the classifier using data from the last period. The importance of the relaxation and comparison of relaxation strategies is presented in Section 6.3.

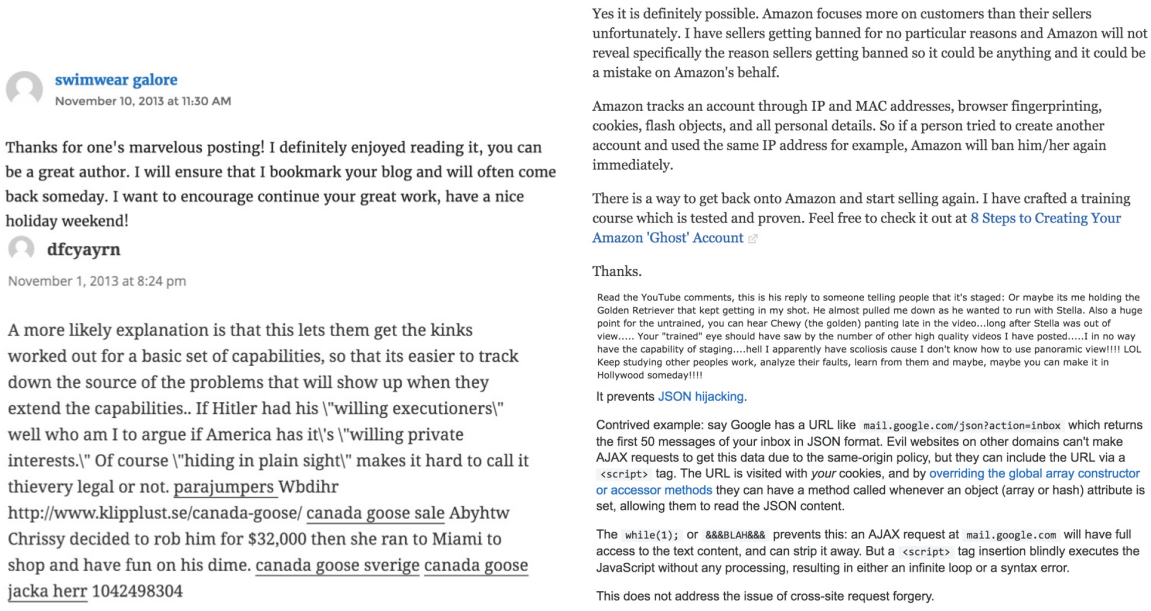
4. Tests

4.1. System Implementation. An implementation of the automaton described in Section 2 needs three classifiers. Each classifier solves another classification issue and should be implemented by a specialised algorithm.

The first classifier works in the initial phase when only a few examples represent one of the classes. A good candidate is a One-Class Support Vector Machine (OSVM) [43], which is an implementation of an SVM classifier [44] for a single class.

Assume that $x_i \in R^d$ are vectors of features of training elements, for $i = 1, 2, \ldots, N$ and for N being the cardinality of the training set. Assume also $K(x, x')$ is a kernel function. Then a OSVM decision function is implemented as

$$f(x) = \operatorname{sgn}\left(\sum_{i=1}^{N} \alpha_i * K(x, x_i) - \rho\right) \quad (5)$$



(a) Spam (b) Nonspam

Figure 4: The examples of comments.

where α_i and ρ are obtained by maximization of the following convex quadratic programming (QP) problem:

$$\frac{1}{2}\left(\sum_{i=1}^{N}\alpha_i * K\left(x, x_i\right)\right)^2 - \rho - \sum_{i=1}^{N}\alpha_i\left(\left(\sum_{i=1}^{N}\alpha_i * K\left(x, x_i\right)\right) - \rho\right) \tag{6}$$

with constrains:

$$\bigwedge_{i\in\{1,2,\ldots,N\}} 0 \leqslant \alpha_i \leqslant \frac{1}{\nu N} \wedge \sum_{j=1}^{N}\alpha_i = 1 \tag{7}$$

where ν is an equivalent of the regularization coefficient C in the binary SVM classification. Homenda et al. [42] compared OSVM with other SVM techniques applied in recognition with rejection.

The second classifier works with imbalanced sets. The RUSBoost algorithm [45] is dedicated to discriminate imbalanced classes. The algorithm creates an ensemble of weak classifiers similarly as bagging techniques. However, in the created learning set the number of examples from the majority class is reduced to reach a given percentage of the minority class.

The last classifier works using full knowledge of data from the previous periods to classify data from the current period. The classifier should provide high accuracy and be a quick learner. An excellent candidate is Random Forest [46]. The algorithm generates many replicas of the training dataset and grows decision trees on them. The final classification decision is a derivation of an ensemble of the created weak classifiers.

All three described classifiers were implemented for the tests.

4.2. Datasets. The data was collected from June 2013 to February 2014 and divided into ten monthly periods labelled T_i where $i = 0 \ldots 9$. The first period T_0 is unique because the preceding period does not exist and it is not possible to use any previous data to create a classifier.

During each period T_i, the system collects two datasets. The first set exS^+ contains web spam comments collected by a spam trap. Figure 4(a) presents examples of web spam. The examples are comments on WordPress blogs. The comments are diversified. The first comment looks like a valid comment but hides a link in the username. The second comment contains starts as normal text and ends with several injected links that do not resemble the normal text anymore. This spam was created to promote link farms and provide credibility to the spammer website.

The second set, labelled as nonspam, consists of nonspam comments. This set is heterogeneous and contains nonspam comments from the three web communities: Quora, Reddit, and Stack Overflow. Examples of comments are presented in Figure 4(b). In the test, the comments from one web service are treated as an internal dataset inS^- when the other two are treated as separate external datasets exS^-.

Quora dataset consists of the best answers to the most popular questions posted on Quora.com. Quora does not provide an API to the rating system. Therefore, a web scraping method was chosen. A bot started crawling from pages with most followed topics in 2014 (https://www.quora.com/What-were-the-most-followed-topics-on-Quora-in-2014) and in 2015 (https://www.quora.com/What-were-the-most-followed-topics-on-Quora-in-2015). Next, the bot collected 105 links to top topic pages.

On each topic page, the bot visited an overviewed top answers and FAQ page to extract the total of 2804 links to

Table 2: Distribution of data among testing periods.

	T_0	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9	Average
Spam trap	2588	7673	7371	4176	1783	7746	17419	15323	14112	2065	3648
Quora	45	40	41	53	59	47	54	53	73	49	23
Reddit	105	362	51	555	319	40	24	867	626	343	150
Stack Overflow	1104	951	972	1028	1057	1007	849	717	884	689	421

individual question pages. On each question page, up to 5 top answers were extracted and saved, for a total of 9520 answers.

The Reddit JSON API was used to collect nonspam comments for Reddit dataset. The bot started with top topics page (http://www.reddit.com/top.json?sort=top&t=all) and entered each topic in the order. On each topic page, all comments were examined. Comments were considered nonspam when the following conditions were met:

(i) A comment was parsed correctly according to Reddit API docs.

(ii) A comment was ranked positively; it has 5 more thumbs-up than thumbs-down.

(iii) A comment was not too short; it has at least 100 characters.

(iv) A comment was not reported as offensive by any user.

When feed was collected a total of 6521 topic pages were visited, and 529158 comments were parsed. Among them, 130604 were rejected because they had the ups/downs balance lower than 5. Among all the comments, 176688 comments were considered too short (less than 100 characters). Finally, 221866 were collected and included in the feed data.

The Stack Exchange API was used to download all answers to top-rated questions on Stack Overflow. When feed has collected a list of 68410 top rated questions was downloaded via API and a total of 500000 answers were extracted and saved. Answers with an upvoted score greater or equal to 30 were selected for a total of 101161 highly rated answers.

The collected comments were limited to comments that overlap the monthly periods when the spam comments were collected. The number of data in the division on data sources and periods and the average monthly volume is given in Table 2.

4.3. Features Extraction. In the preprocessing, before calculating actual features, each analysed comment was transformed into three separate forms to calculate the features proposed in our previous work [11].

The first form was a Visible Text. The HTML document was stripped of all mark-up using the BeatifulSoup4 library with lxml backend. In the result, we obtained the pure text between tags. The second form was a Nonblank Visible Text. To obtain this form, we removed all space characters from the Visible Text. The third created form was Distinct Domains. The Distinct Domains is a set of unique domain names including the domains defined by Internationalized Domain Names in Application (IDNA) standards [47], which were in a language-specific script or alphabet, such as Arabic, Chinese, Russian, or the Latin alphabet-based characters with diacritical marks, such as Polish.

Table 3: Groups of extracted features.

Group	Count
HTML tags features	10
Metadata section features	6
Domains features	7
Global text statistics	12
Statistics for lexical items	22
Alphanumeric and non-alphanumerics characters statistics	6
Total	63

Table 3 presents groups of features used to describe the analysed comments. We have analysed HTML tags and metadata section, domains present in the message, global text features, lexical items, and alphanumeric and non-alphanumerics characters.

The features in the groups based mostly on count and length of described objects in all created forms. Simple statistics such as the average, maximum, and standard deviation were calculated. In summary, we created 63 features.

4.4. Evaluation. The following measures were used during the tests described in Section 5:

TP (true positive) the number of correctly recognised spam entries.

TN (true negative) the number of correctly recognised nonspam entries.

FP (false positive) the number of incorrectly recognised spam entries.

FN (false negative) the number of incorrectly recognised nonspam entries.

Accuracy the fraction of correctly recognised entries (both spam and nonspam)

$$\mathrm{ACC} = \frac{TP + TN}{TP + FP + TN + FN}. \tag{8}$$

Sensitivity or True Positive Rate the fraction of correctly recognised spam entries among all spam entries

$$\mathrm{TPR} = \frac{TP}{TP + FN}. \tag{9}$$

Table 4: The comparison of results obtained at the WEBSPAM-UK data set. The best result from the reference works is presented.

Data set	Measure	Work										
		Our	[11]	[3]	[12]	[13]	[17]	[18]	[19]	[20, 21]	[4]	[22]
WEBSPAM-UK 2006	ACC	0.80	0.78	0.82	-	-	-	-	-	0.88	-	-
	TPR	0.79	0.86	-	-	-	-	-	-	-	-	-
	SPC	0.81	0.69	-	-	-	-	-	-	-	-	-
	F1	0.80	-	0.80	0.81	0.86	-	-	0.95	0.76	-	0.75
WEBSPAM-UK 2007	ACC	0.94	0.92	-	-	-	-	0.92	-	-	-	-
	TPR	1.00	0.96	-	-	-	-	-	-	-	-	-
	SPC	0.10	0.29	-	-	-	-	0.05	-	-	-	-
	F1	0.55	-	-	0.33	0.40	0.41	-	0.44	-	0.69	-

Specificity the fraction of detected nonspam entries among all nonspam entries

$$\mathrm{SPC} = \frac{TN}{TN + FP}. \tag{10}$$

F-measure the weighted average of the spam predictive value and sensitivity,

$$\mathrm{F1} = \frac{2TP}{2TP + FP + FN}. \tag{11}$$

The comparison of web spam recognition mechanisms [25] shows that the F-measure (11) is the most popular evaluator. Therefore, the F-measure is used instead of Area under the Curve (AUC) that we calculated in our previous work [11].

5. Results

5.1. Comparison with Other Works. For a preliminary evaluation of the proposed method, we compared our classifier with selected works mentioned in the newest list of web spam detectors presented in [25]. Among all works, we selected ones that used data from the WEBSPAM-UK repository and evaluated the results using some measures from Section 4.4.

Table 4 compares the results obtained on data from the WEBSPAM-UK repository. We always present the best result obtained by the authors.

For our tests, we created a set of 600 thousand pages from 2006 and took all data from 2007. The data was evenly divided into the learning and testing sets. Luckner at al. [11] used the same dataset and an SVM as a classifier. Despite the higher TPR in 2006 and SPC in 2007 than in this work, the currently obtained accuracy is better.

The other compared works use the same repository, but the division into learning and testing sets was different than in this work. Therefore the results are hard to compare. Specifically, the better results obtained in [20, 21] were computed using tenfold cross-validation. We should remember that although such a test allows us to compare solutions, in practice, we should expect worse results when the training and testing sets are collected in different periods.

Shengen at al. [3] obtained the better results on a smaller testing set contained 204 spam hosts and 300 regular hosts. The obtained F-measure was in the range 0.64-0.73 for an SVM and 0.72-0.80 for genetics operators. Our results are better than most of these results.

Several other works obtained a better F-measure than we on data from 2006. However, in works [12, 13, 22] the best results were obtained for a vector of over 200 features. When we compare the results for the vectors of similar length to us the obtained F-measure is smaller than 0.63. Therefore, one may prefer our method to limit the number of observed features. Although the difference in size of the feature vector does not play a significant role during the classification process, it can increase the training time significantly. Also, the storage issue may be relevant when we want to create a historical learning set.

Our results on WEBSPAM-UK2007 are highly satisfactory. Except for work [4] where tenfold cross-validation was used, we obtained the best results.

However, the quality of the web spam detector decreases over time. Therefore, the rest of the tests were performed on separate online data collected in the consecutive monthly periods.

5.2. Tests on Online Data. We considered ten monthly periods from T_0 to T_9. During the periods both spam comments and nonspam comments were collected. The nonspam comments S^- were collected from three separate sources denoted as $\mathbb{Q}$ (Quora), $\mathbb{R}$ (Reddit), and $\mathbb{S}$ (Stack Overflow). The spam comments S^+ were taken from the spam trap.

In the test, the learning set was always a subset of data from period T_i while the testing set was a subset of data from the following period T_{i+1}. We tested scenarios that represent the various lifecycle phases of the web spam detection system.

(1) When the rejector was in state R_0 during period T_0, the only existing learning set was a set with web spam examples exS_0^+. A One-Class SVM classifier was trained to reject web spam cases among the set $inS_1^+ \cup inS_1^-$.

(2) When some nonspam example comments had already been labelled in period T_0 the rejector switched to state R_1. A RUSBoost algorithm was trained on the set $in_k S_0^- \subset inS_0^- \cup exS_0^-$, where k is the number of labelled nonspam comments. The algorithm rejects web spam from the set $inS_1^+ \cup inS_1^-$.

Table 5: The accuracy obtained by the rejection system during 10 periods.

Testing S^-	Learning S^-	$T_0 \mid R_0$	$T_0 \mid R_1$	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9
$inS^-_{i+1} \in \mathbb{Q}$	$inS^-_i \in \mathbb{Q}$	**0.90**	**1.00**	**1.00**	0.98	**0.99**	**0.96**	**1.00**	**0.99**	**0.99**	**1.00**	**1.00**
$inS^-_{i+1} \in \mathbb{Q}$	$exS^-_i \in \mathbb{R} \cup \mathbb{S}$	**0.90**	0.99	0.95	0.98	0.96	0.89	0.96	**0.99**	**0.99**	0.99	0.99
$inS^-_{i+1} \in \mathbb{Q}$	$S^-_i \in \mathbb{Q} \cup \mathbb{R} \cup \mathbb{S}$	**0.90**	0.98	0.95	**0.99**	0.96	0.90	0.96	0.97	**0.99**	0.99	0.99
$inS^-_{i+1} \in \mathbb{R}$	$inS^-_i \in \mathbb{R}$	**0.90**	0.97	**0.99**	0.98	**0.98**	**0.94**	**0.98**	**1.00**	**0.99**	**1.00**	**1.00**
$inS^-_{i+1} \in \mathbb{R}$	$exS^-_i \in \mathbb{Q} \cup \mathbb{S}$	**0.90**	**1.00**	0.96	0.98	0.96	0.92	0.97	0.97	**0.99**	0.98	0.99
$inS^-_{i+1} \in \mathbb{R}$	$S^-_i \in \mathbb{Q} \cup \mathbb{R} \cup \mathbb{S}$	**0.90**	**1.00**	0.96	**0.99**	0.97	0.90	0.96	0.99	**0.99**	0.99	0.99
$inS^-_{i+1} \in \mathbb{S}$	$inS^-_i \in \mathbb{S}$	**0.91**	**1.00**	0.94	0.98	0.95	**0.93**	**0.97**	**0.98**	0.98	0.98	**0.99**
$inS^-_{i+1} \in \mathbb{S}$	$exS^-_i \in \mathbb{Q} \cup \mathbb{R}$	**0.91**	**1.00**	**0.99**	0.98	**0.97**	0.91	0.95	**0.98**	0.98	0.98	0.94
$inS^-_{i+1} \in \mathbb{S}$	$S^-_i \in \mathbb{Q} \cup \mathbb{R} \cup \mathbb{S}$	**0.91**	**1.00**	0.94	**0.99**	0.96	0.91	0.96	0.97	**0.99**	**0.99**	**0.99**

Table 6: The sensitivity obtained by the rejection system during 10 periods.

Testing S^-	Learning S^-	$T_0 \mid R_0$	$T_0 \mid R_1$	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9
$inS^-_{i+1} \in \mathbb{Q}$	$inS^-_i \in \mathbb{Q}$	**1.00**	0.80	**1.00**	**1.00**	**1.00**	0.97	0.98	0.91	0.98	0.97	0.98
$inS^-_{i+1} \in \mathbb{Q}$	$exS^-_i \in \mathbb{R} \cup \mathbb{S}$	**1.00**	**0.83**	**1.00**	**1.00**	**1.00**	**1.00**	0.98	0.97	0.97	**0.99**	**0.99**
$inS^-_{i+1} \in \mathbb{Q}$	$S^-_i \in \mathbb{Q} \cup \mathbb{R} \cup \mathbb{S}$	**1.00**	**0.83**	**1.00**	**1.00**	**1.00**	**1.00**	**0.99**	**0.98**	**0.99**	**0.99**	**0.99**
$inS^-_{i+1} \in \mathbb{R}$	$inS^-_i \in \mathbb{R}$	**0.97**	0.77	**1.00**	**1.00**	**1.00**	**1.00**	0.97	**1.00**	**0.97**	0.99	0.99
$inS^-_{i+1} \in \mathbb{R}$	$exS^-_i \in \mathbb{Q} \cup \mathbb{S}$	**0.97**	0.99	0.98	0.99	0.97	0.99	0.95	**1.00**	0.98	0.99	0.98
$inS^-_{i+1} \in \mathbb{R}$	$S^-_i \in \mathbb{Q} \cup \mathbb{R} \cup \mathbb{S}$	**0.97**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**0.97**	**1.00**	**1.00**	**1.00**	**1.00**
$inS^-_{i+1} \in \mathbb{S}$	$inS^-_i \in \mathbb{S}$	**0.98**	0.99	**1.00**	**1.00**	**1.00**	**1.00**	**0.99**	**0.99**	**0.99**	**1.00**	**1.00**
$inS^-_{i+1} \in \mathbb{S}$	$exS^-_i \in \mathbb{Q} \cup \mathbb{R}$	**0.98**	**1.00**	0.97	0.95	0.92	0.87	0.73	0.64	0.63	0.74	0.79
$inS^-_{i+1} \in \mathbb{S}$	$S^-_i \in \mathbb{Q} \cup \mathbb{R} \cup \mathbb{S}$	**0.98**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**0.99**	0.98	**0.99**	0.99	**1.00**

(3) During periods T_1 to T_9 there was full knowledge of the spam and nonspam examples from the previous periods. However, the learning nonspam set was limited to the last m examples $in_m S^-_i \subset inS^-_i \cup exS^-_i$. A random forest algorithm was trained on set $in_m S^-_i \cup exS^+_i$ to reject spam among examples from the next period $inS^-_{i+1} \cup inS^+_{i+1}$.

Table 5 presents the accuracy obtained by the rejection system during ten periods. The results are presented individually for each source of nonspam comments. For each source, the results are presented for the internal and external learning sets and their union. The external sets were created using the power set of the nonspam sets $\mathbb{Q}$, $\mathbb{R}$, and $\mathbb{S}$ excluding the internal set. The table shows the average accuracy. The best result for the testing set and period is in bold.

The results for the period T_0 are presented in two columns. The first column presents the results obtained in state R_0. The second column shows the results obtained in state R_1. The transition between states R_0 and R_1 depends on coefficient k in rule r_0. The coefficient was fixed at 20 to stabilise the learning set accuracy (see Section 6.1).

For the following periods, the rejector was in state R_2 and the random forest was used to reject the spam. According to rule r_2, the training set size should be limited by coefficient m. The coefficient was taken as the average number of messages collected in one period (see Table 2). Tables 5, 6, 7, and 8 have the same structure.

Table 5 shows that the accuracy obtained by the rejection system exceeded 0.90 in the initial phase for the learning set $in_k S^-_0$. In the following periods, the classifier trained on inS^-_i obtained results from 0.93 to 1.00 depending on the period and data source. The average accuracy reached 0.99, 0.98, and 0.96 for $\mathbb{Q}$, $\mathbb{R}$, and $\mathbb{S}$, respectively. However, the best accuracy was not always obtained using the internal learning set. A more detail detailed discussion of external sets usage is presented in Section 6.2

Table 6 shows an interesting fact in the initial phase. The classifier trained in state R_0 obtains a better result than the second one trained in state R_1. We should remember that the number of nonspam in the initial phase is relatively small. Therefore, the One-Class SVM classifier could obtain excellent results in detection for the dominant class. In the following periods, the sensitivity, for the internal learning set, is very high for sets $\mathbb{R}$ and $\mathbb{S}$ and varies from 0.97 to 1.00. For the set $\mathbb{Q}$ the sensitivity is lower and varies from 0.91 to 0.97. This could be connected with the smaller cardinal number of this set.

Table 7 presents the specificity. The specificity obtained in the initial phase is similar for all testing sets. The classifiers worked on the same spam set and the number of nonspam examples is small and did not influence the final results. In the following periods, the specificity varies more. The average specificity—calculated on periods for the internal learning set—reaches 0.99, 0.98, and 0.95 for $\mathbb{Q}$, $\mathbb{R}$, and $\mathbb{S}$ sets, respectively. Once again, the worst result for the set $\mathbb{S}$ is connected with the number of nonspam examples in the testing set. For the set $\mathbb{S}$, the number of nonspam examples is relatively higher than for the other sets.

Finally, Table 8 presents the F-measure, which is the weighted average of the spam predictive value and the

Table 7: The specificity obtained by the rejection system during 10 periods.

Testing S^-	Learning S^-	$T_0 \mid R_0$	$T_0 \mid R_1$	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9
$inS^-_{i+1} \in \mathbb{Q}$	$inS^-_i \in \mathbb{Q}$	**0.90**	**1.00**	**1.00**	0.98	**0.99**	**0.96**	**1.00**	**0.99**	**0.99**	**1.00**	**1.00**
$inS^-_{i+1} \in \mathbb{Q}$	$exS^-_i \in \mathbb{R} \cup \mathbb{S}$	**0.90**	**1.00**	0.95	0.98	0.96	0.89	0.96	0.99	**0.99**	0.99	0.99
$inS^-_{i+1} \in \mathbb{Q}$	$S^-_i \in \mathbb{Q} \cup \mathbb{R} \cup \mathbb{S}$	**0.90**	**1.00**	0.95	**0.99**	0.96	0.89	0.96	0.97	**0.99**	0.99	0.99
$inS^-_{i+1} \in \mathbb{R}$	$inS^-_i \in \mathbb{R}$	**0.90**	**1.00**	**0.99**	0.98	**0.98**	**0.93**	**0.98**	**1.00**	**0.99**	**1.00**	**1.00**
$inS^-_{i+1} \in \mathbb{R}$	$exS^-_i \in \mathbb{Q} \cup \mathbb{S}$	**0.90**	**1.00**	0.95	0.98	0.96	0.91	0.97	0.97	**0.99**	0.99	0.99
$inS^-_{i+1} \in \mathbb{R}$	$S^-_i \in \mathbb{Q} \cup \mathbb{R} \cup \mathbb{S}$	**0.90**	**1.00**	0.95	**0.99**	0.96	0.88	0.96	0.99	**0.99**	0.99	0.99
$inS^-_{i+1} \in \mathbb{S}$	$inS^-_i \in \mathbb{S}$	**0.90**	**1.00**	0.93	0.98	0.94	0.89	0.96	0.98	0.98	0.98	0.99
$inS^-_{i+1} \in \mathbb{S}$	$exS^-_i \in \mathbb{Q} \cup \mathbb{R}$	**0.90**	**1.00**	**0.99**	0.98	**0.99**	**0.94**	**0.98**	**1.00**	**0.99**	**1.00**	**1.00**
$inS^-_{i+1} \in \mathbb{S}$	$S^-_i \in \mathbb{Q} \cup \mathbb{R} \cup \mathbb{S}$	**0.90**	**1.00**	0.93	**0.99**	0.95	0.86	0.96	0.97	**0.99**	0.99	0.99

Table 8: The F-measure obtained by the rejection system during 10 periods.

Testing S^-	Learning S^-	$T_0 \mid R_0$	$T_0 \mid R_1$	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9
$inS^-_{i+1} \in \mathbb{Q}$	$inS^-_i \in \mathbb{Q}$	**0.95**	0.90	**1.00**	0.99	**1.00**	**0.96**	**0.99**	0.95	**0.99**	**0.99**	**0.99**
$inS^-_{i+1} \in \mathbb{Q}$	$exS^-_i \in \mathbb{R} \cup \mathbb{S}$	**0.95**	0.91	0.98	0.99	0.98	0.95	0.97	**0.98**	0.98	**0.99**	**0.99**
$inS^-_{i+1} \in \mathbb{Q}$	$S^-_i \in \mathbb{Q} \cup \mathbb{R} \cup \mathbb{S}$	**0.95**	**0.92**	0.98	**1.00**	0.98	0.95	0.98	0.97	**0.99**	**0.99**	**0.99**
$inS^-_{i+1} \in \mathbb{R}$	$inS^-_i \in \mathbb{R}$	**0.94**	0.88	**0.99**	0.99	**0.99**	**0.96**	**0.98**	**1.00**	0.98	**0.99**	**1.00**
$inS^-_{i+1} \in \mathbb{R}$	$exS^-_i \in \mathbb{Q} \cup \mathbb{S}$	**0.94**	0.99	0.97	0.99	0.97	0.95	0.96	0.98	0.98	**0.99**	0.98
$inS^-_{i+1} \in \mathbb{R}$	$S^-_i \in \mathbb{Q} \cup \mathbb{R} \cup \mathbb{S}$	**0.94**	**1.00**	0.98	**1.00**	0.98	0.94	0.97	0.99	**0.99**	**0.99**	**1.00**
$inS^-_{i+1} \in \mathbb{S}$	$inS^-_i \in \mathbb{S}$	**0.94**	0.99	0.97	**0.99**	**0.97**	**0.94**	**0.98**	**0.98**	**0.99**	**0.99**	**0.99**
$inS^-_{i+1} \in \mathbb{S}$	$exS^-_i \in \mathbb{Q} \cup \mathbb{R}$	**0.94**	**1.00**	**0.98**	0.97	0.96	0.90	0.85	0.82	0.81	0.87	0.90
$inS^-_{i+1} \in \mathbb{S}$	$S^-_i \in \mathbb{Q} \cup \mathbb{R} \cup \mathbb{S}$	**0.94**	**1.00**	0.97	**0.99**	**0.97**	0.93	**0.98**	0.97	**0.99**	**0.99**	**0.99**

sensitivity. In the initial phases, the first classifier reaches about 0.94 and the second varies from 0.88 to 0.99. It is the influence of the sensitivity, which is visible in Table 6. However, results in the following periods are similar and stay on a high level for all internal sets. The results vary from 0.94 to 1.00. The differences between the extremes for the various datasets are not higher than 0.02.

6. Discussion

This section presents a more detailed discussion of the following issues. Section 6.1 discusses the results for the initial period. Section 6.2 summarises the usage of external datasets as the learning set. Finally, Section 6.3 discusses the need for system relaxation.

6.1. Initial Period. During the initial period T_0, the system does not possess the full characteristics of either the web spam inS^+ or the nonspam comments inS^- from the previous period. However, to create any classifier, we have to present a representation of at least one of the classes.

Because the historical web spam datasets exist—spam collected before the system start-up or spam from public repositories such as WEBSPAM-UK [7]—we assume that a set exS^+_0 can be created.

We have used the set exS^+_0 to create a One-Class SVM classifier. The results obtained by the classifier are promising. The accuracy of the system exceeds 0.90. Moreover, both the sensitivity and the specificity stay at a similarly high level.

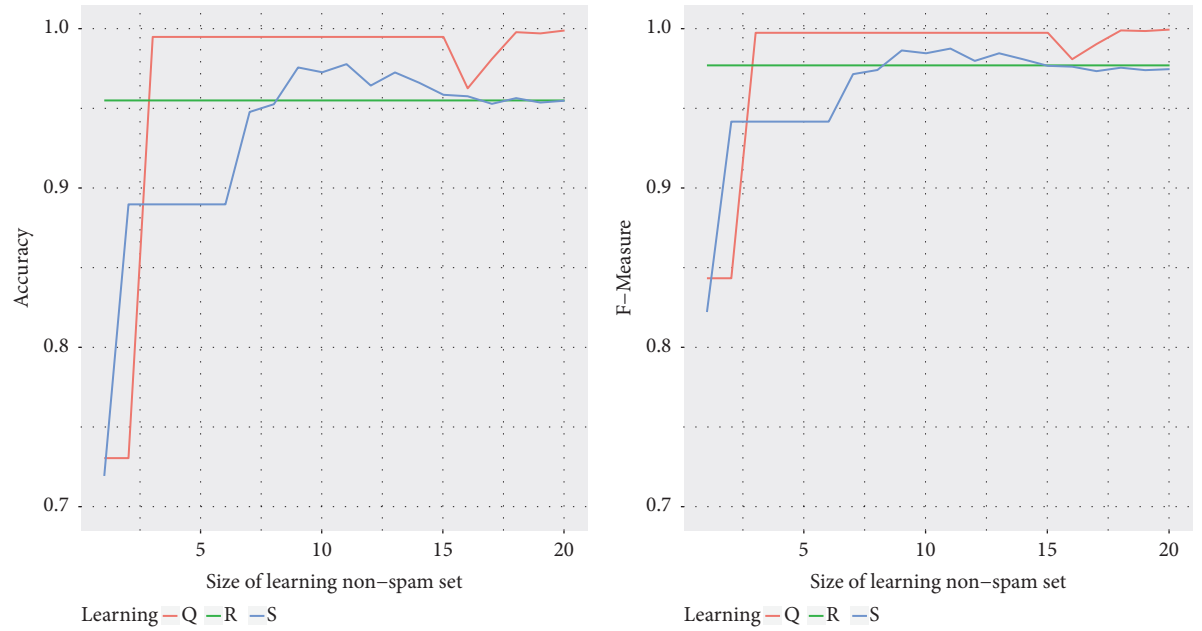
However, to optimise the classification process, the knowledge on web spam characteristic should be supplemented by a partial knowledge of the nonspam comments. This knowledge is represented by a set inS^-_k that contains k examples of the comments. Especially for a small value of k the sets are imbalanced. Therefore, a dedicated classifier—such as RUSBoost—should be used as a discriminator.

Figure 5 presents the accuracy and F-measure obtained by the RUSBoost algorithm on the set $inS^+_1 \cup inS^-_1$ using the learning set $inS^+_0 \cup in_kS^-_0$ (the whole information on spam from the previous period and k nonspam comments).

The accuracy and F-measure obtained on the testing set stabilise fast for the datasets $\mathbb{Q}$ and $\mathbb{R}$. The set $\mathbb{S}$ displays higher diversity and reaches a similar accuracy as $\mathbb{Q}$ and $\mathbb{R}$ for $k > 12$.

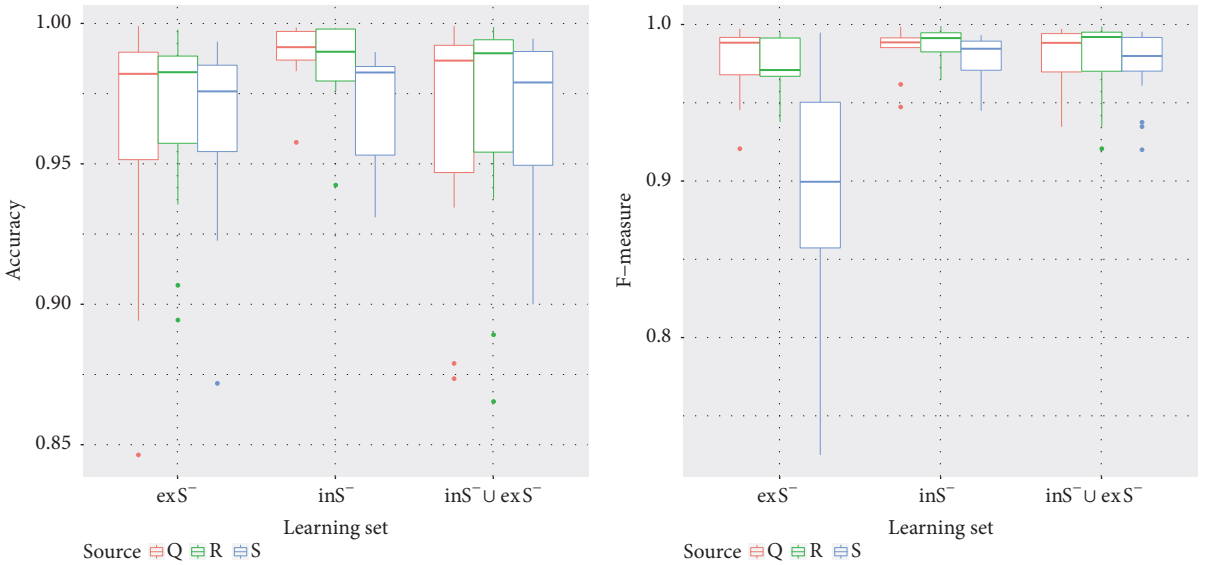
We have compared the results obtained by One-Class SVM trained on inS^+_0 and the results obtained by RUSBoost using $inS^+_0 \cup in_{20}S^-_0$. The RUSBoost algorithm obtained the best accuracy. The difference was up to 9 percentage points. However, we should notice that in some cases the RUSBoost algorithm reduces the obtained F-measure. This is because of the lower sensitivity obtained in several cases. To sum up, the usage of the additional set $in_kS^-_0$ in the learning process is recommended, but the obtained results are not better in all aspects.

6.2. External Dataset as Learning Set. Using an external dataset exS^-_i that contains information on nonspam data one can eliminate the initial period issue. The historical data from



(a) Accuracy (b) F-measure

Figure 5: Results for RusBoost algorithm with limited learning set.



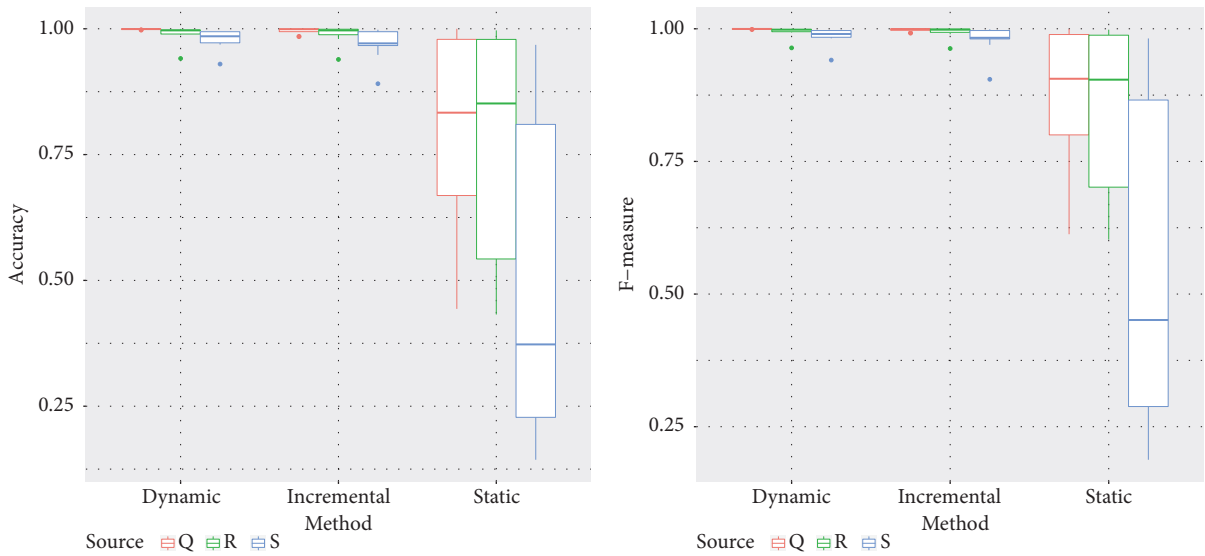
(a) Accuracy (b) F-measure

Figure 6: Results obtained using internal and external nonspam sources.

the external set exS_i^- in combination with spam data from the spam traps exS_i^+ create the complete learning set. However, the external set can reduce the accuracy of the classification process. To check the influence of the external dataset, we analysed the accuracy and F-measure in all periods except T_0. The data was analysed separately for external and internal data.

Figure 6 shows the distribution for three groups. The first group contains the results obtained using inS_i^- as the learning set. The second group consists of results obtained using only external data exS_i^-. The last group contains learning data from both sets $inS_i^- \cup exS_i^-$. The learning sets were created separately for each testing set $\mathbb{Q}$, $\mathbb{R}$, and $\mathbb{S}$.

The best results were obtained using inS_i^- as the learning set. Figure 6(a) shows that additional knowledge of nonspam messages from other sources does not increase the accuracy. However, the main aim of the analysis was to determinate whatever an external source can be used as the learning set.



(a) Accuracy (b) F-measure

Figure 7: Results for static, incremental, and dynamic learning sets.

The results obtained using exS_i^- confirm that. Except for the outliers the distributions for inS_i^- and exS_i^- are similar.

The situation looks different when we analyse F-measure. Figure 6(b) shows that when the results for sources $\mathbb{Q}$ and $\mathbb{R}$ are similar the results for $\mathbb{S}$ are entirely different. In this case, we should remember that for specific web services such as Stack Overflow ($\mathbb{S}$) the usage of external datasets may reduce the obtained sensitivity (see Table 6). Therefore, the system should preferably be used to protect web services with a more conventional form of comments.

The results show that the whole classification process can be fully automated. The average accuracy for all inS^- sets is 0.98. The same measure for the sets exS^+ is 0.96, similarly to the average F-measure that reaches 0.98 and 0.96 for the internal and external sets, respectively.

Therefore, using spam data from the spam traps exS^+ and comments obtained from a public API from the popular web services exS^- one can create a dynamical classifier that rebuilds itself periodically to improve the accuracy.

6.3. System Relaxation. The proposed system uses relaxation of the learning set and compared this solution with two alternative approaches. First, the learning set can be static [1, 11, 14]. The created classifier was tested on periods T_i for $i = 1 \ldots 9$. Second, the learning set can increase. We simulated that by using data from $\sum_{j=0}^{i-1} T_j$ to train a classifier tested on T_i for $i = 1 \ldots 9$. In the case of our dynamic system, data from T_{i-1} were taken to train a classifier tested on T_i for $i = 1 \ldots 9$.

Figure 7 shows that the results obtained for the static learning set are much worse than for the other approaches.

When we analyse the results for the last period, the average accuracy for the static learning set is 0.70 while the other approaches reach 0.99. Similarly with the F-measure which is 0.70 and nearly 1.00 for the static learning set and other approaches, respectively. This shows clearly that a static learning set cannot be used to create a reliable lifelong solution.

However, comparison of the results for the dynamic and incremental approaches is not so simple. Neither accuracy in Figure 7(a) nor F-measure in Figure 7(b) shows any clear difference between them. To prove that there is a significant difference between the results obtained by the two strategies, we performed Wilcoxon's Signed–Rank test for paired scores [48].

If the accuracy obtained by our strategy is significantly better the test should show that the calculated accuracy is higher than for the incremental learning set strategy in most of the tests and smaller in a few tests by only a small amount. We compared the two strategies in all 27 combinations of datasets and periods. For 16 pairs the accuracy calculated for our strategy was greater. The opposite situation occurred in 8 cases. In the rest of the cases, the results were the same for both strategies.

Wilcoxon's Signed–Rank test rejected the null hypothesis ($p = 0.084$), which stated that the results obtained by the two strategies were not significantly different, at the 0.1 level. Moreover, the modified test accepted ($p = 0.044$), at the 0.1 level, the alternate hypothesis that the difference in the accuracy between our strategy and the alternative strategy and the incremental strategy come from a distribution with median greater than 0.

A similar test on 27 combinations of datasets and periods was performed for F-measure. For 13 pairs the F-measure calculated for our strategy was greater. The opposite situation occurred in 8 cases. In the rest of the cases, the results were the same for both strategies.

Wilcoxon's Signed–Rank test rejected the null hypothesis ($p = 0.099$), which stated that the results obtained by the two strategies were not significantly different, at the 0.1 level. Moreover, the modified test accepted ($p = 0.051$), at the

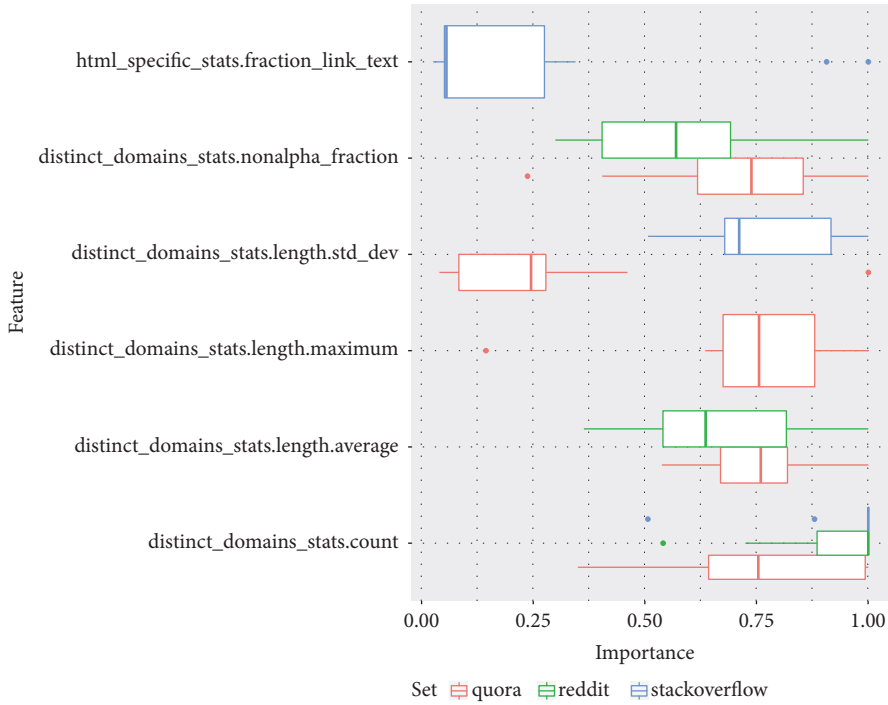


Figure 8: Variation of the importance of features among various periods.

0.1 level, the alternate hypothesis that the difference in F1 between our strategy and the alternative strategy come from a distribution with median greater than 0.

Therefore, the results obtained by our strategy are significantly better than the results obtained by the other strategies when the strategies are evaluated using the accuracy and the F-measure.

Let us discuss the approach that uses sliding time windows for training data. Such an approach is proposed in several works, e.g., [15, 29, 30]. The approach looks natural but is not as stable as our method. In the tests of the dynamic approach, we rebuild the learning set when the size of learning classes exceeds some fixed value m. Therefore, the classifier is always trained on a similar structure of the learning sets. Using the time window we work with changing classes and the ratio between them. That influences classification quality as it shows the comparison between the dynamic and the incremental learning sets. We should stress that optimisation of coefficient m is still an open issue, which could be the subject of future works that will enhance the advantages of the proposed strategy.

Finally, let us discuss why the idea of the static learning set fails. It is because the mathematical definition of spam and nonspam changes over time in an unforeseen way. Therefore, the predictions become less accurate as time passes. To prove this reasoning, we tested changes of features importance in the classification process.

An estimation of predictor importance for decision trees was calculated. Feature importance is calculated for a split defined by the given feature. Importance is computed as the difference between the Mean Squared Error (MSE) for the parent node and the total MSE for the two children in the regression task. In the classification task, the Gini coefficient is used instead to estimate how the data space in the node is divided among classes. The Gini coefficient equals $2(AUC) - 1$, where AUC is the area underneath the Receiver Operating Characteristic Curve (ROC Curve).

For a random forest, the used function computes predictor importance for all weak learners. For every decision tree, the sum of changes in the MSE is calculated due to splits on every feature used in the recognition process. Next, the sum is divided by the number of branch nodes.

Importance is normalised to the range $[0, 1]$ with 0 representing the smallest possible importance.

Figure 8 shows the distribution of the normalised importance over time. For clarity, we have limited the number of presented features to the set of features that obtained importance 1 for some sets. Each distribution was calculated for each set separately. The influence of the other features on the classification results may still be important, but we can limit the current discussion to the essential features.

The distribution shows that the essential feature in one period can be less critical in other periods. Therefore, it is not possible to create a classifier using a static learning set that obtains a stable accuracy of web spam detection.

7. Conclusions

In this paper, we have proposed an intelligent machine learning system for web spam detection. As an engine of the system, we proposed an automaton that creates a reliable lifelong machine learning solution by switching classification mechanisms according to the current learning set (see Section 3.2).

The proposed solution can protect newly created web pages. In the initial phase—when nonspam messages typical for the web page are not well known—the system uses dedicated algorithms to protect the pages with accuracy that exceeds 0.9 (see Table 5). In the subsequent stages, the system stabilises on 0.96-0.99 depending on the data source (see Table 5).

The system does not rely on a static learning set. The built-in mechanism collects data on new web spam received from the spam traps as well as on nonspam comments from third-party Web 2.0 platforms (see Section 3.3).

This mechanism forms a fully automated high-level protection system. We have compared the results of spam classification obtained using learning sets consisting of data from the protected system and from the external sources. In the first case, when an operator must label the nonspam examples, the accuracy reached 0.98 and F-measure 0.98. The fully automated system based on external data achieved accuracy and F-measure of 0.96 and 0.96, respectively (see Figure 6).

In comparison to the static learning set approach, the results obtained by the system were better by nearly 0.3 both for the accuracy and F-measure (see Figure 7). Also, it was proved by a statistical test that the proposed solution is significantly better than the approach based on an incrementally extended learning set (see Section 6.3).

All elements of the classification process were tested on real data from spam traps and common known web services: Quora, Reddit, and Stack Overflow. The obtained average accuracy over time was 0.99, 0.98, and 0.96, respectively (see Table 5). This shows that the system is immune to accuracy failure over time and that the proposed solution can be used successfully to protect real services against web spam.

Future works will focus on increasing automation of the system by estimation of the parameters of the transition function in the automation. The changes should pay off in the optimisation of the qualification results and higher stability of the system.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The author declares no conflicts of interest.

Acknowledgments

The data was collected using Antyscam system that was developed in EU POIG.01.04.00-14-031/11 Project. The research was supported by the National Science Center, Grant no. 2012/07/B/ST6/01501, decision no. UMO–2012/07/B/ST6/01501.

References

[1] M. Erdélyi, A. A. Benczúr, J. Masanés, and D. Siklósi, "Web spam filtering in internet archives," in *Proceedings of the 5th International Workshop on Adversarial Information Retrieval on the Web*, pp. 17–20, ACM, Madrid, Spain, 2009.

[2] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: an update," *ACM SIGKDD Explorations Newsletter*, vol. 11, no. 1, pp. 10–18, 2009.

[3] L. Shengen, N. Xiaofei, L. Peiqi, and W. Lin, "Generating new features using genetic programming to detect link spam," in *Proceedings of the 4th International Conference on Intelligent Computation Technology and Automation (ICICTA)*, vol. 1, pp. 135–138, IEEE Computer Society, Shenzhen, China, March 2011.

[4] M. Mahmoudi, A. Yari, and S. Khadivi, "Web spam detection based on discriminative content and link features," in *Proceedings of the 2010 5th International Symposium on Telecommunications*, pp. 542–546, IEEE, Tehran, Iran, December 2010.

[5] S. P. Algur and N. T. Pendari, "Hybrid spamicity score approach to web spam detection," in *Proceedings of the 2012 International Conference on Pattern Recognition, Informatics and Medical Engineering*, pp. 36–40, IEEE, Salem, India, March 2012.

[6] C. Dong and B. Zhou, "Effectively detecting content spam on the web using topical diversity measures," in *Proceedings of the 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, vol. 1, pp. 266–273, IEEE Computer Society, Macau, China, December 2012.

[7] I. Bíró, D. Siklósi, J. Szabó, and A. A. Benczúr, "Linked latent dirichlet allocation in web spam filtering," in *Proceedings of the 5th International Workshop on Adversarial Information Retrieval on the Web*, pp. 37–40, ACM, Madrid, Spain, April 2009.

[8] G. V. Cormack, M. D. Smucker, and C. L. A. Clarke, "Efficient and effective spam filtering and re-ranking for large web datasets," *Information Retrieval*, vol. 14, no. 5, pp. 441–465, 2011.

[9] M. Erdélyi, A. Garzó, and A. A. Benczúr, "Web spam classification: a few features worth more," in *Proceedings of the 2011 Joint WICOW/AIRWeb Workshop on Web Quality*, pp. 27–34, ACM, Hyderabad, India, March 2011.

[10] K. L. Goh, R. K. Patchmuthu, and A. K. Singh, "Link-based web spam detection using weight properties," *Journal of Intelligent Information Systems*, vol. 43, no. 1, pp. 129–145, 2014.

[11] M. Luckner, M. Gad, and P. Sobkowiak, "Stable web spam detection using features based on lexical items," *Computers & Security*, vol. 46, pp. 79–93, 2014.

[12] J. Martinez-Romo and L. Araujo, "Web spam identification through language model analysis," in *Proceedings of the 5th International Workshop on Adversarial Information Retrieval on the Web*, pp. 21–28, ACM, Madrid, Spain, April 2009.

[13] L. Araujo and J. Martínez-Romo, "Web spam detection: new classification features based on qualified link analysis and language models," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 3, pp. 581–590, 2010.

[14] K. L. Goh, A. K. Singh, and K. H. Lim, "Multilayer perceptrons neural network based Web spam detection application," in *Proceedings of the 2013 IEEE China Summit and International Conference on Signal and Information Processing*, pp. 636–640, IEEE, Beijing, China, July 2013.

[15] R. Colbaugh and K. Glass, "Predictive defense against evolving adversaries," in *Proceedings of the 2012 10th IEEE International Conference on Intelligence and Security Informatics*, pp. 18–23, IEEE, Arlington, Tex, USA, June 2012.

[16] M. Brückner and T. Scheffer, "Nash equilibria of static prediction games," in *Proceedings of the 22nd International Conference*

on Neural Information Processing Systems, Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, Eds., pp. 171–179, Curran Associates Inc., Vancouver, Canada, 2009.

[17] N. Dai, B. D. Davison, and X. Qi, "Looking into the past to better classify web spam," in *Proceedings of the 5th International Workshop on Adversarial Information Retrieval on the Web*, pp. 1–8, ACM, Madrid, Spain, April 2009.

[18] F. Asdaghi and A. Soleimani, "An effective feature selection method for web spam detection," *Knowledge-Based Systems*, vol. 166, pp. 198–206, 2019.

[19] R. C. Patil and D. R. Patil, "Web spam detection using SVM classifier," in *Proceedings of the 2015 IEEE 9th International Conference on Intelligent Systems and Control (ISCO)*, pp. 1–4, IEEE, Coimbatore, India, January 2015.

[20] L. Becchetti, C. Castillo, D. Donato, R. Baeza-Yates, and S. Leonardi, "Link-based characterization and detection of web spam," in *Proceedings of the 2nd International Workshop on Adversarial Information Retrieval on the Web*, vol. 2, pp. 1–8, SIGIR, Seattle, Wash, USA, 2006.

[21] C. Castillo, D. Donato, A. Gionis, V. Murdock, and F. Silvestri, "Know your neighbors: web spam detection using the web topology," in *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, W. Kraaij, A. P. de Vries, C. L. A. Clarke, N. Fuhr, and N. Kando, Eds., pp. 423–430, ACM, Amsterdam, The Netherlands, July 2007.

[22] R. K. Roul, S. R. Asthana, M. Shah, and D. Parikh, "Detecting spam web pages using content and link-based techniques," *Sādhanā*, vol. 41, no. 2, pp. 193–202, 2016.

[23] A. Heydari, M. A. Tavakoli, N. Salim, and Z. Heydari, "Detection of review spam: a survey," *Expert Systems with Applications*, vol. 42, no. 7, pp. 3634–3642, 2015.

[24] J. Mathew, A. K. Singh, and K. L. Goh, "Proceedings of the 4th international conference on eco-friendly computing and communication systems comprehensive literature review on machine learning structures for web spam classification," *Procedia Computer Science*, vol. 70, pp. 434–441, 2015.

[25] A. Makkar and N. Kumar, "Cognitive spammer: a framework for pagerank analysis with split by over-sampling and train by under-fitting," *Future Generation Computer Systems*, vol. 90, pp. 381–404, 2019.

[26] X.-C. Yin, K. Huang, C. Yang, and H.-W. Hao, "Convex ensemble learning with sparsity and diversity," *Information Fusion*, vol. 20, no. 1, pp. 49–59, 2014.

[27] B. Manaskasemsak and A. Rungsawang, "Web spam detection using trust and distrust-based ant colony optimization learning," *International Journal of Web Information Systems*, vol. 11, no. 2, pp. 142–161, 2015.

[28] C. Castillo, D. Donato, L. Becchetti et al., "A reference collection for web spam," *ACM SIGIR Forum*, vol. 40, no. 2, pp. 11–24, 2006.

[29] S. Hao, A. Kantchelian, B. Miller, V. Paxson, and N. Feamster, "PREDATOR: Proactive recognition and elimination of domain abuse at time-of-registration," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1568–1579, ACM, Vienna, Austria, October 2016.

[30] K. Veeramachaneni, I. Arnaldo, V. Korrapati, C. Bassias, and K. Li, "AI2: training a big data machine to defend," in *Proceedings of the 2016 IEEE 2nd International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS)*, pp. 49–54, IEEE, New York, NY, USA, April 2016.

[31] A. Alarifi and M. Alsaleh, "Web spam: a study of the page language effect on the spam detection features," in *Proceedings of the 2012 11th International Conference on Machine Learning and Applications*, vol. 2, pp. 216–221, IEEE, Boca Raton, FLa, USA, December 2012.

[32] E. Georgiou, M. D. Dikaiakos, and A. Stassopoulou, "On the properties of spam-advertised URL addresses," *Journal of Network and Computer Applications*, vol. 31, no. 4, pp. 966–985, 2008.

[33] Z. Sadan and D. G. Schwartz, "Social network analysis of web links to eliminate false positives in collaborative anti-spam systems," *Journal of Network and Computer Applications*, vol. 34, no. 5, pp. 1717–1723, 2011, Dependable Multimedia Communications: Systems, Services, and Applications.

[34] T. Urvoy, E. Chauveau, P. Filoche, and T. Lavergne, "Tracking Web spam with HTML style similarities," *ACM Transactions on the Web (TWEB)*, vol. 2, no. 1, pp. 3:1–3:28, 2008.

[35] J. Piskorski, M. Sydow, and D. Weiss, "Exploring linguistic features for web spam detection: A preliminary study," in *Proceedings of the 4th international workshop on Adversarial information retrieval on the web*, pp. 25–28, ACM, Beijing, China, April 2008.

[36] R. Islam and J. Abawajy, "A multi-tier phishing detection and filtering approach," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 324–335, 2013.

[37] S. M. Lee, D. S. Kim, J. H. Kim, and J. S. Park, "Spam detection using feature selection and parameters optimization," in *Proceedings of the 2010 International Conference on Complex, Intelligent and Software Intensive Systems*, pp. 883–888, IEEE, Krakow, Poland, 2010.

[38] E. Dolzhenko, J. Ligatti, and S. Reddy, "Modeling runtime enforcement with mandatory results automata," *International Journal of Information Security*, vol. 14, no. 1, pp. 47–60, 2014.

[39] S. H. Seyyedi and B. Minaei-Bidgoli, "Estimator learning automata for feature subset selection in high-dimensional spaces, case study: Email spam detection," *International Journal of Communication Systems*, vol. 31, no. 8, p. e3541, 2018.

[40] J. Fdez-Glez, D. Ruano-Ordas, J. R. Méndez, F. Fdez-Riverola, R. Laza, and R. Pavón, "A dynamic model for integrating simple web spam classification techniques," *Expert Systems with Applications*, vol. 42, no. 21, pp. 7969–7978, 2015.

[41] J. Fdez-Glez, D. Ruano-Ordás, R. Laza, J. R. Méndez, R. Pavón, and F. Fdez-Riverola, "WSF2: a novel framework for filtering web spam," *Scientific Programming*, vol. 2016, Article ID 6091385, 18 pages, 2016.

[42] W. Homenda, M. Luckner, and W. Pedrycz, "Classification with rejection based on various SVM techniques," in *Proceedings of the 2014 International Joint Conference on Neural Networks (IJCNN)*, pp. 3480–3487, IEEE, Beijing, China, 2014.

[43] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

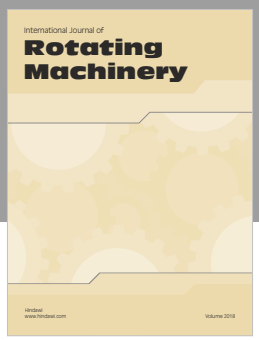
[44] V. N. Vapnik, *The Nature of Statistical Learning Theory*, Springer-Verlag, New york, NY, USA, 1995.

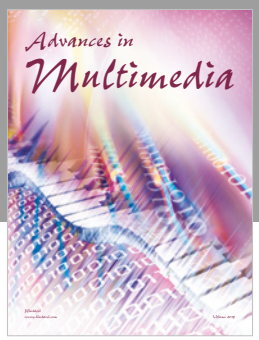
[45] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "RUSBoost: A hybrid approach to alleviating class imbalance," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 40, no. 1, pp. 185–197, 2010.

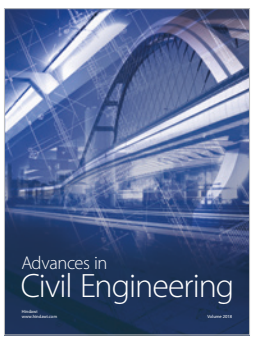
[46] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

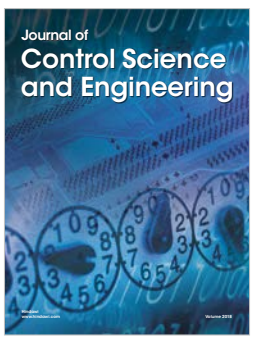
[47] P. Faltstrom, P. Hoffman, and A. Costello, "Internationalizing domain names in applications (IDNA)," Internet RFC 3490, 2003.

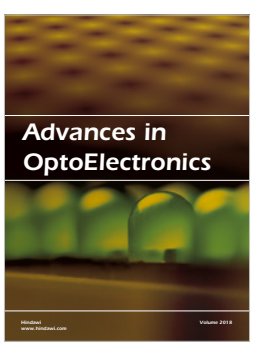
[48] N. Japkowicz and M. Shah, *Evaluating Learning Algorithms: A Classification Perspective*, Cambridge University Press, Cambridge, UK, 2011.

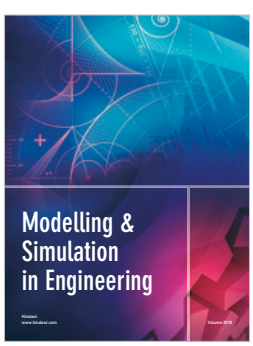


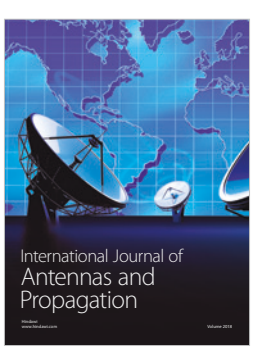


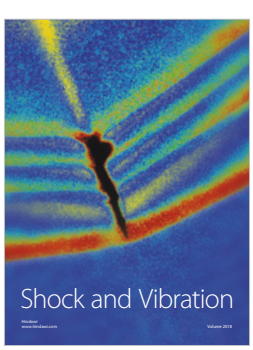


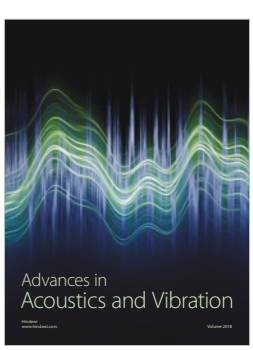












Hindawi

Submit your manuscripts at
www.hindawi.com