The classification power of Web features[*]

Miklós Erdélyi[1,2], András A. Benczúr[1,3], Bálint Daróczy[1,3], András Garzó[1,3], Tamás Kiss[1,3] and Dávid Siklósi[1,3]

[1]Computer and Automation Research Institute, Hungarian Academy of Sciences (MTA SZTAKI)
[2]University of Pannonia, Department of Computer Science and Systems Technology, Veszprém
[3]Eötvös University Budapest

Abstract

In this paper we give a comprehensive overview of features devised for Web spam detection and investigate how much various classes, some requiring very high computational effort, add to the classification accuracy.

- We collect and handle a large number of features based on recent advances in Web spam filtering, including temporal ones, in particular we analyze the strength and sensitivity of linkage change.
- We propose new temporal link similarity based features and show how to compute them efficiently on large graphs.
- We show that machine learning techniques including ensemble selection, LogitBoost and Random Forest significantly improve accuracy.
- We conclude that, with appropriate learning techniques, a simple and computationally inexpensive feature subset outperforms all previous results published so far on our data set and can only slightly be further improved by computationally expensive features.
- We test our method on three major publicly available data sets, the Web Spam Challenge 2008 data set WEBSPAM-UK2007, the ECML/PKDD Discovery Challenge data set DC2010 and the Waterloo Spam Rankings for ClueWeb09.

[*]This work was supported in part by the EC FET Open project "New tools and algorithms for directed network analysis" (NADINE No 288956), by the EU FP7 Project LAWA—Longitudinal Analytics of Web Archive Data, OTKA NK 105645 and by the European Union and the European Social Fund through project FuturICT.hu (grant no.: TAMOP-4.2.2.C-11/1/KONV-2012-0013). The research was carried out as part of the EITKIC_12-1-2012-0001 project, which is supported by the Hungarian Government, managed by the National Development Agency, financed by the Research and Technology Innovation Fund and was performed in cooperation with the EIT ICT Labs Budapest Associate Partner Group. (`www.ictlabs.elte.hu`) This paper is a comprehensive comparison of the best performing classification techniques based on [9, 37, 36, 38] and new experiments.

Our classifier ensemble sets the strongest classification benchmark as compared to participants of the Web Spam and ECML/PKDD Discovery Challenges as well as the TREC Web track.

To foster research in the area, we make several feature sets and source codes public[1], including the temporal features of eight `.uk` crawl snapshots that include WEBSPAM-UK2007 as well as the Web Spam Challenge features for the labeled part of ClueWeb09.

1 Introduction

Web classification finds several use, both for content filtering and for building focused corpora from a large scale Web crawl. As one notable use, Internet archives actively participate in large scale experiments [8], some of them building analytics services over their collections [6]. Most of the existing results on Web classification originate from the area of Web spam filtering that have turned out to generalize to a wide class of tasks including genre, Open Directory category, as well as quality classification. Closely related areas include filtering and tagging in social networks [50].

Web spam filtering, the area of devising methods to identify useless Web content with the sole purpose of manipulating search engine results, has drawn much attention in the past years [63, 49, 46]. The first mention of Web spam, termed *spamdexing* as a combination of words *spam* and (search engine) *indexing*, appears probably in a 1996 news article [27] as part of the early Web era discussions on the spreading porn content [24]. In the area of the so-called Adversarial Information Retrieval workshop series ran since 2005 [40] and evaluation campaigns including the Web Spam Challenges [18], the ECML/PKDD Discovery Challenge 2010 [50] and the Spam task of TREC 2010 Web Track [29] were organized. A recent comprehensive survey on Web spam filtering research is found in [19].

In this paper we present, to our best knowledge, the most comprehensive experimentation based on content, link as well as temporal features, both new and recently published. Our spam filtering baseline classification procedures are collected by analyzing the results [28, 1, 44] of the Web Spam Challenges and the ECML/PKDD Discovery Challenge 2010 [45, 2, 58]. Our comparison is based on AUC values [42] that we believe to be more stable as it does not depend on the split point; indeed, while Web Spam Challenge 2007 used F-measure and AUC, Web Spam Challenge 2008 used AUC only as evaluation measure.

Web spam appears in sophisticated forms that manipulate content as well as linkage [47] with examples such as

- Copied content, "honey pots" that draw attention but link to unrelated, spam targets;
- Garbage content, stuffed with popular or monetizable query terms and phrases such as university degrees, online casinos, bad credit status or

[1] `https://datamining.sztaki.hu/en/download/web-spam-resources`

adult content;

- Link farms, a large number of strongly interlinked pages across several domains.

The Web spammer toolkit consists of a clearly identifiable set of manipulation techniques that has not changed much recently. The Web Spam Taxonomy of Gyöngyi et al. [47] distinguishes content (term) and link spamming along with techniques of hiding, cloaking and removing traces by e.g. obfuscated redirection. Most of the features designed fight either link or content spamming.

We realize that recent results have ignored the importance of the machine learning techniques and concentrated only on the definition of new features. Also the only earlier attempt to unify a large set of features [20] is already four years old and even there little comparison is given on the relative power of the feature sets. For classification techniques, a wide selection including decision trees, random forest, SVM, class-feature-centroid, boosting, bagging and oversampling in addition to feature selection (Fisher, Wilcoxon, Information Gain) were used [45, 2, 58] but never compared and combined. In this paper we address the following questions.

- Do we get the maximum value out of the features we have? Are we sufficiently sophisticated at applying machine learning?
- Is it worth calculating computationally expensive features, in particular some related to page-level linkage?
- What is an optimal feature set for a fast spam filter that can quickly react at crawl time after fetching a small sample of a Web site?

We compare our result with the very strong baselines of the Web Spam Challenge 2008 and ECML/PKDD 2010 Discovery Challenge data sets. Our main results are as follows.

- We apply state-of-the-art classification techniques by the lessons learned from KDD Cup 2009 [57]. Key in our performance is ensemble classification applied both over different feature subsets as well as over different classifiers over the same features. We also apply classifiers yet unexplored against Web spam, including Random Forest [14] and LogitBoost [43].
- We compile a small yet very efficient feature set that can be computed by sample pages from the site while completely ignoring linkage information. By this feature set a filter may quickly react to a recently discovered site and intercept in time before the crawler would start to follow a large number of pages from a link farm. This feature set itself reaches AUC 0.893 over WEBSPAM-UK2007.
- Last but not least we gain strong improvements over the Web Spam Challenge best performance [18]. Our best result in terms of AUC reaches 0.9 and improves on the best Discovery Challenge 2010 results.

Several recent papers propose temporal features [61, 55, 31, 52] to improve classification accuracy. We extend link-based similarity algorithms by proposing metrics to capture the linkage change of Web pages over time. We describe a method to calculate these metrics efficiently on the Web graph and then measure their performance when used as features in Web spam classification. We propose an extension of two link-based similarity measures: XJaccard and PSimRank [41].

We investigate the combination of temporal and non-temporal, both link- and content-based features using ensemble selection. We evaluate the performance of ensembles built on the latter feature sets and compare our results to that of state-of-the-art techniques reported on our dataset. Our conclusion is that temporal and link-based features in general do not significantly increase Web spam filtering accuracy. However, information about linkage change might improve the performance of a language independent classifier: the best results for the French and German classification tasks of the ECML/PKDD Discovery Challenge [45] were achieved by using host level link features only, outperforming those who used all features [2].

In this paper we address not just the quality but also the computational efficiency. Earlier lightweight classifiers include Webb et al. [64] describing a procedure based solely on the HTTP session information. Unfortunately they only measure precision, recall and F-measure that are hard to compare with later results on Web spam that use AUC. In fact the F and similar measures greatly depend on the classification threshold and hence make comparison less stable and for this reason they are not used starting with the Web Spam Challenge 2008. Furthermore in [64] the IP address is a key feature that is trivially incorporated in the DC2010 data set by placing all hosts from the same IP address into the same training or testing set. The intuition is that if an IP address contains spam hosts, all hosts from that IP address are likely to be spam and should be immediately manually checked and excluded from further consideration.

The rest of this paper is organized as follows. In Section 2 we describe the data sets used in this paper. We give an overview of temporal features for spam detection and propose new temporal link similarity based ones in Section 3. In Section 4 we describe our classification framework. The results of the experiments to classify WEBSPAM-UK2007, ClueWeb09 and DC2010 can be found in Section 5. The computational resource needs of various feature sets are summarized in Section 6.

2 Data Sets

In this paper we use three data sets, WEBSPAM-UK2007 of the Web Spam Challenge 2008 [18], the Waterloo Spam Rankings for ClueWeb09, and DC2010 created for the ECML/PKDD Discovery Challenge 2010 on Web Quality. We only give a brief summary of the first data set described well in [18, 22] and the second in [38], however, we describe the third one in more detail in Section 2.3. Also we compare the amount of spam in the data sets.

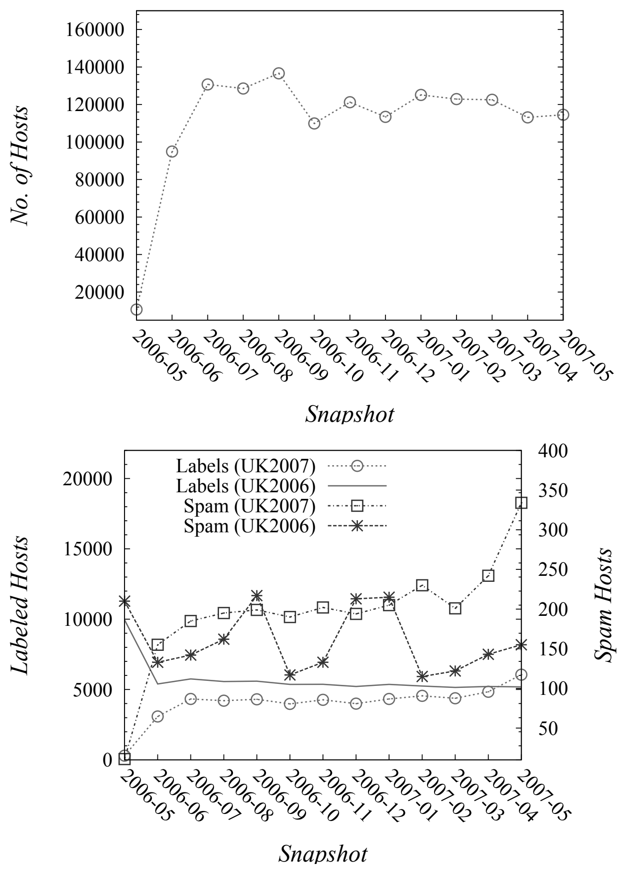

Figure 1: The number of total and labeled hosts in the 13 UK snapshots. We indicate the number of positive and negative labels separate for the WEBSPAM-UK2006 and WEBSPAM-UK2007 label sets.

2.1 Web Spam Challenge 2008: WEBSPAM-UK2007

The Web Spam Challenge was first organized in 2007 over the WEBSPAM-UK2006 data set. The last Challenge over the WEBSPAM-UK2007 set was held in conjunction with AIRWeb 2008 [18]. The Web Spam Challenge 2008 best result [44] achieved an AUC of 0.85 by also using ensemble undersampling [23]. They trained a bagged classifier on the standard content-based and link-based features published by the organizers of the Web Spam Challenge 2008 and on custom host-graph based features, using the ERUS strategy for class-inbalance learning. For earlier challenges, best performances were achieved by a semi-supervised version of SVM [1] and text compression [28]. Best results either used bag of words vectors or the so-called "public" feature sets of [20].

We extended the WEBSPAM-UK2007 data set with 13 `.uk` snapshots pro-

Label Set	Instances	%Positive
Training	4000	5.95%
Testing	2053	4.68%

Table 1: Summary of label sets for Web Spam Challenge 2008.

vided by the Laboratory for Web Algorithmics of the Università degli studi di Milano. We use the training and testing labels of the Web Spam Challenge 2008, as summarized in Table 1. In order to prepare a temporal collection, we extracted maximum 400 pages per site from the original crawls. The last 12 of the above `.uk` snapshots were analyzed by Bordino et al. [12] who observe a relative low URL but high host overlap[2]. The first snapshot (2006-05) that is identical to WEBSPAM-UK2006 was chosen to be left out from their experiment since it was provided by a different crawl strategy. We observed that in the last eight snapshots the number of hosts have stabilized in the sample and these snapshots have roughly the same amount of labeled hosts as seen in Fig. 1. From now on we restrict attention to the aforementioned subset of the snapshots and the WEBSPAM-UK2007 labels only.

2.2 The Waterloo Spam Rankings for ClueWeb09

The English part of ClueWeb09 consist of approximately 20M domains and 500M pages. For Web spam labels we used the Waterloo Spam Rankings [29]. While the Waterloo Spam Rankings contain negative training instances as well, we extended the negative labels with the set of the Open Directory Project (ODP) hosts. We used 50% split for training and testing.

We labeled hosts in both the `.pt` crawl and ClueWeb09 by top-level ODP categories using links extracted from topic subtrees in the directory. Out of all labeled hosts, 642643 received a unique label. Because certain sites (e.g., `bbc.co.uk`) may belong to even all 14 top-level English categories, we discarded the labels of 18734 hosts with multiple labels to simplify the multi-label task. As Bordino et al. [13] indicate, multitopical hosts are often associated to poor quality sites and spam as another reason why their labels may mislead the classification process. The resulting distribution of labels is shown in Table 2.

2.3 Discovery Challenge 2010: DC2010

The Discovery Challenge was organized over DC2010, a new data set that we describe in more detail next. DC2010 is a large collection of annotated Web hosts labeled by the Hungarian Academy of Sciences (English documents), Internet Memory Foundation (French) and L3S Hannover (German). The base data is a set of 23M pages in 190K hosts in the `.eu` domain crawled by the Internet Memory Foundation in early 2010.

[2]The dataset can be downloaded from: `http://law.di.unimi.it/datasets.php`

Category	**No. of Hosts**	**% of Labeled Hosts**
spam	439	0.07%
Arts	97355	15.1%
Business	193678	30.1%
Computers	66159	10.3%
Recreation	65594	10.2%
Science	43317	6.7%
Society	122084	19%
Sports	54456	8.5%

Table 2: Number of positive ClueWeb09 host labels for spam and the ODP categories.

	UK2006	**UK2007**	**ClueWeb09**	**DC2010**			
				en	de	fr	all
Hosts	10 660	114 529	500,000	61 703	29 758	7 888	190 000
Spam	19.8%	5.3%	unknown	8.5% of valid labels; 5% of all in large domains.			

Table 3: Fraction of Spam in WEBSPAM-UK2006, UK2007, ClueWeb09 and DC2010. Note that three languages English, German and French were selected for labeling DC2010, although Polish and Dutch language hosts constitute a larger fraction than the French. Since to our best knowledge, no systematic random sample was labeled for ClueWeb09, the number 439 of labeled spam hosts is not representative for the collection.

The labels extend the scope of previous data sets on Web Spam in that, in addition to sites labeled spam, we included manual classification for genre into five categories Editorial, Commercial, Educational, Discussion and Personal as well as trust, factuality and bias as three aspects of quality. Spam label is exclusive since no other assessment was made for spam. However other labels are non-exclusive and hence define nine binary classification problems. We consider no multi-class tasks in this paper. Assessor instructions are for example summarized in [62], a paper concentrating on quality labels.

In Table 3, we summarize the amount of spam in the DC2010 data set in comparison with the Web Spam Challenge data sets. This amount is well-defined for the latter data sets by the way they were prepared for the Web Spam Challenge participants. However for DC2010, this figure may be defined in several ways. First of all, when creating the DC2010 labels, eventually we considered domains with or without a `www.` prefix the same such as `www.domain.eu` vs. `domain.eu`. However in our initial sampling procedure we considered them as two different hosts and merged them after verifying that the labels of the two versions were identical. Also, several domains consist of a single redirection page to another domain and we counted these domains, too. Finally, a large fraction of spam is easy to spot and can be manually removed. As an example of many

Count	IP address	Comment
3544	80.67.22.146	spam farm `*-palace.eu`
3198	78.159.114.140	spam farm `*auts.eu`
1374	62.58.108.214	`blogactiv.eu`
1109	91.204.162.15	spam farm `x-mp3.eu`
1070	91.213.160.26	spam farm `a-COUNTRY.eu`
936	81.89.48.82	`autobazar.eu`
430	78.46.101.76	spam farm `77k.eu` and 20+ domains
402	89.185.253.73	spam farm `mp3-stazeni-zdarma.eu`

Table 4: Selection of IP addresses with many subdomains in the DC2010 data set.

Label	Group	Yes	Maybe	No
Spam	*Spam*	423		4 982
News/Editorial	*Genre*	191		4 791
Commercial		2 064		2 918
Educational		1 791		3 191
Discussion		259		4 724
Personal-Leisure		1 118		3 864
Non-Neutrality	*Quality*	19	216	3 778
Bias		62		3 880
Dis-Trustiness		26	201	3 786

Table 5: Distribution of assessor labels in the DC2010 data set.

hosts on same IP, we include a labeled sample from DC2010, that itself contains over 10,000 spam domains in Table 4. These hosts were identified by manually looking at the IP addresses that serve the largest number of domain names. Thus our sample is biased and obtaining an estimate of the spam fraction is nontrivial, as indicated in Table 3.

The distribution of labels for the nine categories with more than 1% positive samples (spam, news, commercial, educational, discussion, personal, neutral, biased, trusted) is given in Table 5. For Neutrality and Trust the strong negative categories have low frequency and hence we fused them with the intermediate negative (maybe) category for the training and testing labels.

The Discovery Challenge 2010 best result [58] achieved an AUC of 0.83 for spam classification while the overall winner [45] was able to classify a number of quality components at an average AUC of 0.80. As for the technologies, bag of words representation variants proved to be very strong for the English collection while only language independent features were used for German and French. The applicability of dictionaries and cross-lingual technologies remains open.

New to the construction of the DC2010 training and test set is the handling of hosts from the same domain and IP address. Since no IP address and domain was allowed to be split between training and testing, we might have to reconsider the applicability of propagation [48, 66] and graph stacking [54]. The Web Spam Challenge data sets were labeled by uniform random sampling and graph stacking appeared to be efficient in several results [22] including our prior work [30]. The applicability of graph stacking remains however unclear for the DC2010 data set. Certain teams used some of these methods but reported no improvement [2].

3 Temporal Features for Spam Detection

Spammers often create bursts in linkage and content: they may add thousands or even millions of machine generated links to pages that they want to promote [61] that they again very quickly regenerate for another target or remove if blacklisted by search engines. Therefore changes in both content and linkage may characterize spam pages.

Recently the evolution of the Web has attracted interest in defining features, signals for ranking [34] and spam filtering [61, 55, 31, 52, 37]. The earliest results investigate the changes of Web content with the primary interest of keeping a search engine index up-to-date [25, 26]. The decay of Web pages and links and its consequences on ranking are discussed in [4, 35]. One main goal of Boldi et al. [11] who collected the `.uk` crawl snapshots also used in our experiments was the efficient handling of time-aware graphs. Closest to our temporal features is the investigation of host overlap, deletion and content dynamics in the same data set by Bordino et al. [12].

Perhaps the first result on the applicability of temporal features for Web spam filtering is due to Shen et al. [61] who compare pairs of crawl snapshots and define features based on the link growth and death rate. However by extending their ideas to consider multi-step neighborhood, we are able to define a very strong feature set that can be computed by the Monte Carlo estimation of Fogaras and Rácz [41]. Another result defines features based on the change of the content [31] who obtain page history from the Wayback Machine.

For calculating the temporal link-based features we use the host level graph. As observed in [12], pages are much more unstable over time compared to hosts. Note that page-level fluctuations may simply result from the sequence the crawler visited the pages and not necessarily reflect real changes. The inherent noise of the crawling procedure and problems with URL canonization [5] rule out the applicability of features based on the change of page-level linkage.

3.1 Linkage Change

In this section we describe link-based temporal features that capture the extent and nature of linkage change. These features can be extracted from either the page or the host level graph where the latter has a directed link from host a to host b if there is a link from a page of a to a page of b.

The starting point of our new features is the observation of [61] that the in-link growth and death rate and change of clustering coefficient characterize the evolution patterns of spam pages. We extend these features for the multi-step neighborhood in the same way as PageRank extends the in-degree. The *ℓ-step neighborhood* of page v is the set of pages reachable from v over a path of length at most ℓ. The ℓ-step neighborhood of a host can be defined similarly over the host graph.

We argue that the changes in the multi-step neighborhood of a page should be more indicative of the spam or honest nature of the page than its single-step neighborhood because spam pages are mostly referred to by spam pages [21], and spam pages can be characterized by larger change of linkage when compared to honest pages [61].

In the following we review the features related to linkage growth and death from [61] in Section 3.1.1, then we introduce new features based on the similarity of the multi-step neighborhood of a page or host. We show how the XJaccard and PSimRank similarity measure can be used for capturing linkage change in Section 3.1.3 and Section 3.1.4, respectively.

3.1.1 Change Rate of In-links and Out-links

We compute the following features introduced by Shen et al. [61] on the host level for a node a for graph instances from time t_0 and t_1. We let $G(t)$ denote the graph instance at time t and $I^{(t)}(a)$, $\Gamma^{(t)}(a)$ denote the set of in and out-links of node a at time t, respectively.

- In-link death (IDR) and growth rate (IGR):

$$\mathrm{IDR}(a) = \frac{|I^{(t_0)}(a)| - |I^{(t_0)}(a) \cap I^{(t_1)}(a)|}{|I^{(t_0)}(a)|}$$

$$\mathrm{IGR}(a) = \frac{|I^{(t_1)}(a)| - |I^{(t_0)}(a) \cap I^{(t_1)}(a)|}{|I^{(t_0)}(a)|}$$

- Out-link death and growth rates (ODR, OGR): the above features calculated for out-links;
- Mean and variance of IDR, IGR, ODR and OGR across in-neighbors of a host (IDRMean, IDRVar, etc.);
- Change rate of the clustering coefficient (CRCC), i.e. the fraction of linked hosts within those pointed by pairs of edges from the same host:

$$CC(a,t) = \frac{|\{(b,c) \in G(t) | b, c \in \Gamma^{(t)}(a)|}{|\Gamma^{(t)}(a)|}$$

$$CRCC(a) = \frac{CC(a,t_1) - CC(a,t_0)}{CC(a,t_0)}$$

- Derivative features such as the ratio and product of the in and out-link rates, means and variances. We list the in-link derivatives; out-link ones are defined similarly:

IGR·IDR, IGR/IDR, IGRMean/IGR, IGRVar/IGR, IDRMean/IDR, IDRVar/IDR, IGRMean · IDRMean, IGRMean/IDRMean, IGRVar · IDRVar, IGRVar/IDRVar.

3.1.2 Self-Similarity Along Time

In the next sections we introduce new linkage change features based on multi-step graph similarity measures that in some sense generalize the single-step neighborhood change features of the previous section. We characterize the change of the multi-step neighborhood of a node by defining the similarity of a single node *across* snapshots instead of two nodes within a single graph instance. The basic idea is that, for each node, we measure its similarity to itself in two identically labeled graphs representing two consecutive points of time. This enables us to measure the linkage change occurring in the observed time interval using ordinary graph similarity metrics.

First we describe our new contribution, the extension of two graph similarity measures, XJaccard and PSimRank [41] to capture temporal change; moreover, we argue why SimRank [51] is inappropriate for constructing temporal features.

SimRank of a pair of nodes u and v is defined recursively as the average similarity of the neighbors of u and v:

$$\begin{aligned}
\mathrm{Sim}_{\ell+1}(u,v) &= 0, \text{ if } I(u) \text{ or } I(v) \text{ is empty;} \\
\mathrm{Sim}_{\ell+1}(u,v) &= 1, \text{ if } u = v; \\
\mathrm{Sim}_{\ell+1}(u,v) &= \frac{c}{|I(u)||I(v)|} \sum_{\substack{v' \in I(v) \\ u' \in I(u)}} \mathrm{Sim}_{\ell}(u',v'),
\end{aligned} \tag{1}$$

where $I(x)$ denotes the set of vertices linking to x and $c \in (0,1)$ is a decay factor. In order to apply SimRank for similarity of a node v between two snapshots t_0 and t_1, we apply (2) so that v' and u' are taken from different snapshots.

Next we describe a known deficiency of SimRank in its original definition that rules out its applicability for temporal analysis. First we give the example for the single graph SimRank. Consider a bipartite graph with k nodes pointing all to another two u and v. In this graph there are no directed paths of length more than one and hence the Sim values can be computed in a single iteration. Counter-intuitively, we get $\mathrm{Sim}(u,v) = c/k$, i.e. the larger the cocitation of u and v, the smaller their SimRank value. The reason is that the more the number of in-neighbors, the more likely is that a pair of random neighbors will be different.

While the example of the misbehavior for SimRank is somewhat artificial in the single-snapshot case, next we show that this phenomenon almost always happens if we consider the similarity of a single node v across two snapshots. If there is no change at all in the neighborhood of node v between the two snapshots, we expect the Sim value to be maximal. However the situation is identical to the bipartite graph case and Sim will be inversely proportional to the number of out-links.

3.1.3 Extended Jaccard Similarity Along Time

Our first definition of similarity is based on the extension of the Jaccard coefficient in a similar way XJaccard is defined in [41]. The Jaccard similarity of a page or host v across two snapshots t_0 and t_1 is defined by the overlap of its neighborhood in the two snapshots, $\Gamma^{(t_0)}(v)$ and $\Gamma^{(t_1)}(v)$ as

$$\mathsf{Jac}^{(t_0,t_1)}(v) = \frac{|\Gamma^{(t_0)}(v) \cap \Gamma^{(t_1)}(v)|}{|\Gamma^{(t_0)}(v) \cup \Gamma^{(t_1)}(v)|}$$

The *extended Jaccard coefficient, XJaccard* for length ℓ of a page or host is defined via the notion of the neighborhood $\Gamma_k^{(t)}(v)$ at distance exactly k as

$$\mathsf{XJac}_\ell^{(t_0,t_1)}(v) = \sum_{k=1}^{\ell} \frac{|\Gamma_k^{(t_0)}(v) \cap \Gamma_k^{(t_1)}(v)|}{|\Gamma_k^{(t_0)}(v) \cup \Gamma_k^{(t_1)}(v)|} \cdot c^k(1-c),$$

where c is a decay factor.

The XJac values can be approximated by the min-hash fingerprinting technique for Jaccard coefficients [15], as described in Algorithm 3 of [41]. The fingerprint generation algorithm has to be repeated for each graph snapshot, with the same set of independent random permutations.

We generate temporal features based on the XJac values for four length values $\ell = 1 \ldots 4$. We also repeat the computation on the transposed graph, i.e. replacing out-links $\Gamma^{(t)}(v)$ by in-links $I^{(t)}(v)$. As suggested in [41], we set the decay factor $c = 0.1$ as this is the value where, in their experiments, XJaccard yields best average quality for similarity prediction.

Similar to [61], we also calculate the mean and variance $\mathsf{XJac}^{(t_0,t_1)}{}_\ell(w)$ of the neighbors w for each node v. The following derived features are also calculated:

- similarity at path length $\ell = 2, 3, 4$ divided by similarity at path length $\ell - 1$, and the logarithm of these;
- logarithm of the minimum, maximum, and average of the similarity at path length $\ell = 2, 3, 4$ divided by the similarity at path length $\ell - 1$.

3.1.4 PSimRank Along Time

Next we define similarity over time based on PSimRank, a SimRank variant defined in [41] that can be applied similar to XJaccard in the previous section. As we saw in Section 3.1.2, SimRank is inappropriate for measuring linkage change in time. In the terminology of the previous subsection, the reason is that path fingerprints will be unlikely to meet in a large neighborhood and SimRank values will be low even if there is completely no change in time.

We solve the deficiency of SimRank by allowing the random walks to meet with higher probability when they are close to each other: a pair of random walks at vertices u', v' will advance to the same vertex (i.e., meet in one step) with probability of the Jaccard coefficient $\frac{|I(u') \cap I(v')|}{|I(u') \cup I(v')|}$ of their in-neighborhood $I(u')$ and $I(v')$.

The random walk procedure corresponding to PSimRank along with a fingerprint generation algorithm is defined in [41].

For the temporal version, we choose independent random permutations σ_ℓ on the hosts for each step ℓ. In step ℓ if the random walk from vertex u is at u', it will step to the in-neighbor with smallest index given by the permutation σ_ℓ in each graph snapshot.

Temporal features are derived from the PSimRank similarity measure very much the same way as for XJaccard, for four length values $\ell = 1 \dots 4$. We also repeat the computation on the transposed graph, i.e. replacing out-links $\Gamma^{(t)}(v)$ by in-links $I^{(t)}(v)$. As suggested in [41], we set the decay factor $c = 0.15$ as this is the value where, in their experiments, PSimRank yields best average quality for similarity prediction. Additionally, we calculate the mean and variance PSimRank(w) of the neighbors w for each node v and derived features as for XJaccard.

3.2 Content and its Change

The content of Web pages can be deployed in content classification either via statistical features such as entropy [59] or via term weight vectors [67, 31]. Some of the more complex features that we do not consider in this work include language modeling [3].

In this section we focus on capturing term-level changes over time. For each target site and crawl snapshot, we collect all the available HTML pages and represent the site as the bag-of-words union of all of their content. We tokenize content using the ICU library[3], remove stop words[4] and stem using Porter's method.

We treat the resulting term list as the virtual document for a given site at a point of time. As our vocabulary we use the most frequent 10,000 terms found in at least 10% and at most 50% of the virtual documents.

To measure the importance of each term i in a virtual document d at time snapshot T, we use the BM25 weighting [60]:

$$t_{i,d}^{(T)} = \mathrm{IDF}_i^{(T)} \cdot \frac{(k_1+1)\mathrm{tf}_{i,d}^{(T)}}{K + \mathrm{tf}_{i,d}^{(T)}}$$

where $\mathrm{tf}_{i,d}^{(T)}$ is the number of occurrences of term i in document d and $\mathrm{IDF}_i^{(T)}$ is the inverse document frequency (Robertson-Spärck Jones weight) for the term at time T. The length normalized constant K is specified as

$$k_1((1-b) + b \times \mathrm{dl}^{(T)}/\mathrm{avdl}^{(T)})$$

such that $dl^{(T)}$ and $avdl^{(T)}$ denote the virtual document length and the average length over all virtual documents at time T, respectively. Finally

$$\mathrm{IDF}^{(T)} = log \frac{N - n^{(T)} + 0.5}{n^{(T)} + 0.5}$$

[3]http://icu-project.org/

[4]http://www.lextek.com/manuals/onix/stopwords1.html

where N denotes the total number of virtual documents and $n^{(T)}$ is the number of virtual documents containing term i. Note that we keep N independent of T and hence if document d does not exist at T, we consider all $\mathrm{tf}_{i,d}^{(T)} = 0$.

By using the term vectors as above, we calculate the temporal content features described in [31] in the following five groups.

- **Ave:** Average BM25 score of term i over the $T_{\max}$ snapshots:

$$\mathrm{Ave}_{i,d} = \frac{1}{T_{\max}} \cdot \sum_{T=1}^{T_{\max}} t_{i,d}^{(T)}$$

- **AveDiff:** Mean difference between temporally successive term weight scores:

$$\mathrm{AveDiff}_{i,d} = \frac{1}{T_{\max}-1} \cdot \sum_{T=1}^{T_{\max}-1} |t_{i,d}^{(T+1)} - t_{i,d}^{(T)}|$$

- **Dev:** Variance of term weight vectors at all time points:

$$\mathrm{Dev}_{i,d} = \frac{1}{T_{\max}-1} \cdot \sum_{T=1}^{T_{\max}} (t_{i,d}^{(T)} - \mathrm{Ave}_{i,d})^2$$

- **DevDiff:** Variance of term weight vector differences of temporally successive virtual documents:

$$\mathrm{DevDiff}_{i,d} = \frac{1}{T_{\max}-2} \cdot \sum_{T=1}^{T_{\max}-1} (|t_{i,d}^{(T+1)} - t_{i,d}^{(T)}| - \mathrm{AveDiff}_{i,d})^2$$

- **Decay:** Weighted sum of temporally successive term weight vectors with exponentially decaying weight. The base of the exponential function, the *decay rate* is denoted by λ. **Decay** is defined as follows:

$$\mathrm{Decay}_{i,d} = \sum_{T=1}^{T_{\max}} \lambda e^{\lambda(T_{\max}-T)} t_{i,d}^{(T)}$$

4 Classification Framework

For the purposes of our experiments we computed all the public Web Spam Challenge content and link features of [20]. We built a classifier ensemble by splitting features into related sets and for each we use a collection of classifiers that fit the data type and scale. These classifiers were then combined by ensemble selection. We used the classifier implementations of the machine learning toolkit Weka [65].

Ensemble selection is an overproduce and choose method allowing to use large collections of diverse classifiers [17]. Its advantages over previously published methods [16] include optimization to any performance metric and refinements to prevent overfitting, the latter being unarguably important when more classifiers are available for selection. The motivation for using ensemble selection is that recently this particular ensemble method gained more attention thanks to the winners of KDD Cup 2009 [57]. In our experiments [38] ensemble selection performed significantly better than other classifier combination methods used for Web spam detection in the literature, such as log-odds based averaging [56] and bagging.

In the context of combining classifiers for Web classification, to our best knowledge, ensemble selection has not been applied yet. Previously, only simple methods that combine the predictions of SVM or decision tree classifiers through logistic regression or random forest have been used [28]. We believe that the ability to combine a large number of classifiers while preventing overfitting makes ensemble selection an ideal candidate for Web classification, since it allows us to use a large number of features and learn different aspects of the training data at the same time. Instead of tuning various parameters of different classifiers, we can concentrate on finding powerful features and selecting the main classifier models which we believe to be able to capture the differences between the classes to be distinguished.

We used the ensemble selection implementation of Weka [65] for performing the experiments. Weka's implementation supports the proven strategies to avoid overfitting such as model bagging, sort initialization and selection with replacement. We allow Weka to use all available models in the library for greedy sort initialization and use 5-fold embedded cross-validation during ensemble training and building. We set AUC as the target metric to optimize for and run 100 iterations of the hillclimbing algorithm.

We mention that we have to be careful with treating missing feature values. Since the temporal features are based on at least two snapshots, for a site that appears only in the last one, all temporal features have missing value. For classifiers that are unable to treat missing values we define default values depending on the type of the feature.

4.1 Learning Methods

We use the following models in our ensemble: bagged and boosted decision trees, logistic regression, naive Bayes and variants of random forests. For most classes of features we use all classifiers and let selection choose the best ones. The exception is static and temporal term vector based features where, due to the very large number of features, we may only use Random Forest and SVM. We train our models as follows.

Bagged LogitBoost: we do 10 iterations of bagging and vary the number of iterations from 2 to 64 in multiples of two for LogitBoost.

Decision Trees: we generate J48 decision trees by varying the splitting criterion, pruning options and use either Laplacian smoothing or no smoothing at all.

Bagged Cost-sensitive Decision Trees: we generate J48 decision trees with default parameters but vary the cost sensitivity for false positives in steps of 10 from 10 to 300. We do the same number of iterations of bagging as for LogitBoost models.

Logistic Regression: we use a regularized model varying the ridge parameter between 10^{-8} to 10^{4} by factors of 10. We normalize features to have mean 0 and standard deviation 1.

Random Forests: we use FastRandomForest [39] instead of the native Weka implementation for faster computation. The forests have 250 trees and, as suggested in [14], the number of features considered at each split is $s/2$, s, $2s$, $4s$ and $8s$, where s is the square root of the total number of features available.

Naive Bayes: we allow Weka to model continuous features either as a single normal or with kernel estimation, or we let it discretize them with supervised discretization.

5 Results and Discussion

In this section we describe the various ensembles we built and measure their performance[5]. We compare feature sets by using the same learning methods described in Section 4 while varying the subset of features available for each of the classifier instances when training and combining these classifiers using ensemble selection. We also concentrate on the value of temporal information for Web spam detection. As our goal is to explore the computational cost vs. classification performance tradeoff, we will describe the resource needs for various features in detail in Section 6.

For training and testing we use the official Web Spam Challenge 2008 training and test sets [20]. As it can be seen in Table 1 these show considerable class imbalance which makes the classification problem harder. For DC2010 we also use the official training set as described in Table 5. For ClueWeb09 we used a 50% random split.

To make it easy to compare our results to previous results, we cite the Web Spam Challenge 2008 and Discovery Challenge 2010 winner's performance in the summary tables next. For ClueWeb09 the only previous evaluation is in terms of TREC retrieval performance [29] that we cannot directly compare here.

5.1 Content-only Ensemble

We build three different ensembles over the content-only features in order to assess performance by completely eliminating linkage information. The feature sets available for these ensembles are the following:

- (A) Public content [59, 22] features without any link based information. Features for the page with maximum PageRank in the host are not used to save the PageRank computation. Corpus precision, the fraction of words in a page that is corpuswise frequent and corpus recall, the fraction

[5]The exact classifier model specification files used for Weka and the data files used for the experiments are available upon request from the authors.

of corpuswise frequent terms in the page are not used either since they require global information from the corpus.

- (Aa) The tiniest feature set of 24 features from (A): query precision and query recall defined similar to corpus precision and recall but based on popular terms from a proprietary query log[6] instead of the entire corpus. A very strong feature set based on the intuition that spammers use terms that make up popular queries.
- (B) The full public content feature set [22], including features for the maximum PageRank page of the host.
- Feature set (B) plus a bag of words representation derived from the BM25 [60] term weighting scheme.

Table 6 presents the performance comparison of ensembles built using either of the above feature sets. The DC2010 and ClueWeb09 detailed results are in Table 8 and Table 9, respectively. Performance is given in AUC for all data sets.

Feature Set	Number of Features	UK2007	DC2010	ClueWeb09
Content (A)	74	0.859	0.757	0.829
Content (Aa)	24	0.841	0.726	0.635
Content (B)	96	0.879	0.799	0.827
BM25 + (B)	10096	**0.893**	**0.891**	**0.870**
Challenge best	-	0.852	0.830	-

Table 6: AUC value of spam ensembles built from content based features.

Surprisingly, with the small (Aa) feature set of only 24 features a performance only 1% worse than that of the Web Spam Challenge 2008 winner can be achieved who employed more sophisticated methods to get their result. By using all the available content based features without linkage information, we get roughly the same performance as the best which have been reported on our data set so far. However this achievement can be rather attributed to the better machine learning techniques used than the feature set itself since the features used for this particular measurement were already publicly accessible at the time of the Web Spam Challenge 2008.

As it can be seen in Table 6 relative performance of content based features over different corpora varies a lot. In case of DC2010 and ClueWeb09 the small (Aa) feature set achieves much worse result than the largest feature set having best performance for all data sets. The fact that the content (A, Aa, B) and link (Table 7) performances are always better for UK2007 might be explained

[6] A summary is available as part of our data release at `https://dms.sztaki.hu/sites/dms.sztaki.hu/files/download/2013/enpt-queries.txt.gz`.

by the fact that the UK2007 training and testing sets were produced by random sampling without considering domain boundaries. Hence in a large domain with many subdomains, part of the hosts belong to the training and part to the testing set with very similar distribution. This advantage disappears for the BM25 features.

5.2 Full Ensemble

Feature Set	Number of Features	UK2007	DC2010	ClueWeb09
Public link-based [7]	177	0.759	0.587	0.806
All combined	10 273	**0.902**	0.885	0.876

Table 7: Performance of ensembles built on link based and all features.

Results of the ensemble incorporating all the previous classifiers is seen in Table 7. The DC2010 detailed results are in Table 8. Overall, we observe that BM25 is a very strong feature set that may even be used itself for a lightweight classifier. On the other hand, link features add little to quality and the gains apparently diminish for DC2010, likely due to the fact that the same domain and IP address is not split between training and testing.

The best Web Spam Challenge 2008 participant [44] reaches an AUC of 0.85 while for DC2010, the best spam classification AUC of [58] is 0.83. We outperform these results by a large margin.

For DC2010 we also show detailed performance for nine attributes in Table 8, averaged in three groups: spam, genre and quality (as in Table 5). Findings are similar: with BM25 domination, part or all of the content features slightly increase the performance. Results for the quality attributes and in particular for trust are very low. Classification for these aspects remains a challenging task for the future.

For ClueWeb09 detailed performance for selected ODP categories can be seen in Table 9. Identically to DC2010 results BM25 features provide the best classification performance. However, combinations with other feature sets yield gains only for spam classification. For the ODP classification tasks linkage information does not help in general: the content based feature set has roughly the same performance with or without page-level linkage information, and combining with the link based feature set does not improve performance notably in most labeling tasks.

5.3 Temporal Link Ensembles

First, we compare the temporal link features proposed in Section 3.1 with those published earlier [61]. Then, we build ensembles that combine the temporal with

Feature Set	spam	genre **average**	quality **average**	**average**
Public link-based [7]	0.655	0.614	0.519	0.587
Content (A)	0.757	0.713	0.540	0.660
Content (Aa)	0.726	0.662	0.558	0.634
Content (B)	0.799	0.735	0.512	0.668
BM25	**0.876**	**0.805**	**0.584**	**0.739**
Public link-based + (B)	0.812	0.731	0.518	0.669
BM25 + (A)	0.872	**0.816**	0.580	**0.754**
BM25 + (B)	**0.891**	0.810	**0.612**	0.744
All combined	0.885	0.813	0.553	0.734

Table 8: Performance over the DC2010 labels in terms of AUC.

Feature Set	spam	Arts	Business	Computers	Recreation	Science	Society	Sports	**ODP average**
Link [7]	.806	.569	.593	.591	.532	.624	.540	.504	.595
Content (A)	.829	.676	.726	.632	.669	.720	.639	.673	.695
Content (Aa)	.635	.508	.524	.554	.487	.558	.502	.522	.536
Content (B)	.827	.673	.727	.634	.670	.720	.629	.674	.694
BM25	.845	**.913**	**.890**	**.931**	**.907**	**.883**	**.915**	**.959**	**.914**
Link + (B)	.848	.675	.731	.646	.669	.727	.631	.669	.699
BM25 + (A)	.871	.895	.881	.896	.879	.851	.904	.935	.892
BM25 + (B)	.869	.895	.881	.898	.892	.850	.906	.934	.894
All combined	**.876**	.896	.883	.898	.892	.852	.905	.936	.895

Table 9: Performance over the ClueWeb09 labels in terms of AUC.

the public link-based features described by [7]. The results are summarized in Table 10. Note that all experiments in this section and Section 5.4 were carried out on the WEBSPAM-UK2007 data set.

As these measurements show, our proposed graph similarity based features successfully extend the growth and death rate based ones by achieving higher accuracy, improving AUC by 1.3%. However, by adding temporal to static link-based features we get only marginally better ensemble performance.

To rank the link-based feature sets by their contribution in the ensemble, we build classifier models on the three separate feature subsets (public link-based, growth/death rate based and graph similarity based features, respectively) and let ensemble selection combine them. This restricted combination results in a slightly worse AUC of 0.762. By calculating the total weight contribution,

Section	**Feature Set**	**No. of Features**	**AUC**
3.1.1	Growth/death rates	29	0.617
3.1.3-4	XJaccard + PSimRank	63	0.625
	Public link-based [7]	176	0.765
3.1.1	Public + growth/death rates	205	0.758
3.1.3-4	Public + XJaccard + PSimRank	239	**0.769**
	All link-based	268	0.765
	WSC 2008 Winner	-	0.852

Table 10: Performance of ensembles built on link-based features.

we get the following ranked list (weight contribution showed in parenthesis): public link-based (60.8%), graph similarity based (21.5%), growth/death rate based (17.7%). This ranking also supports the findings presented in Table 10 that graph similarity based temporal link-based features should be combined with public link-based features if temporal link-based features are used.

To separate the effect of ensemble selection on the performance of temporal link-based feature sets we repeat the experiments with bagged cost-sensitive decision trees only, a model reported to be effective for web spam classification [59]. The results for these experiments are shown in Table 11.

Section	**Feature Set**	**No. of Features**	**AUC**
3.1.1	Growth/death rates	29	0.605
3.1.3	XJaccard	42	0.626
3.1.4	PSimRank	21	0.593
3.1.3-4	XJaccard + PSimRank	63	0.610
	Public link-based [7]	176	**0.731**
3.1.1	Public + growth/death rates	205	0.696
3.1.3-4	Public + XJaccard + PSimRank	239	*0.710*
	All link-based	268	0.707
	WSC 2008 Winner	-	0.852

Table 11: Performance of bagged cost-sensitive decision trees trained on link-based features.

As it can be seen in Table 11, when using bagged cost-sensitive decision trees, our proposed temporal link-based similarity features achieve 3.5% better performance than the growth/death rate based features published earlier.

When comparing results in Table 11 and in Table 10 we can see that ensemble selection i) significantly improves accuracy (as expected) and ii) diminishes the performance advantage achievable by the proposed temporal link-based features over the previously published ones.

As evident from Table 11, the proposed PSimRank based temporal features perform roughly the same as the growth and death rate based ones while the XJaccard based temporal features perform slightly better.

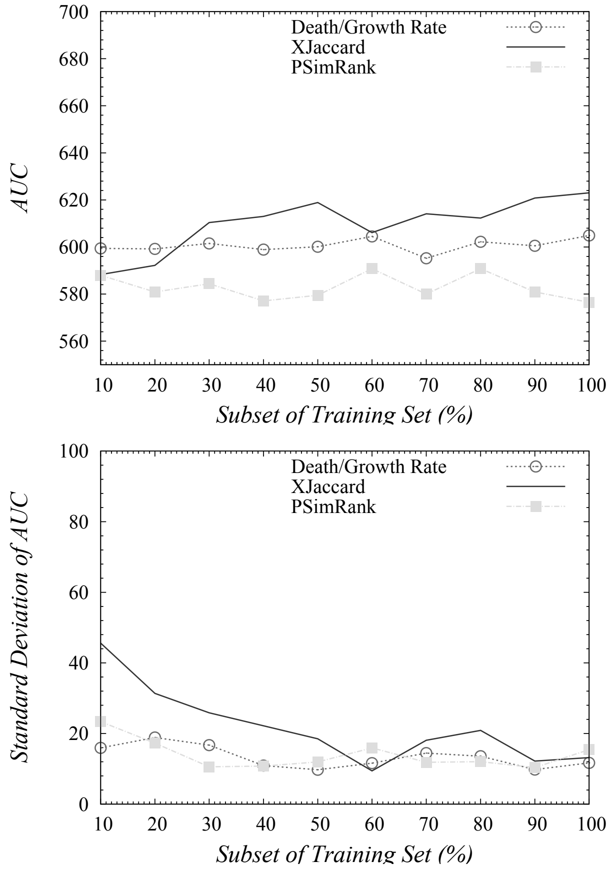

Figure 2: Sensitivity of temporal link-based features. **Top:** AUC values averaged across 10 measurements. **Bottom:** standard deviations of AUC for different training set sizes.

Next we perform sensitivity analysis of the temporal link-based features by using bagged cost-sensitive decision trees. We build 10 different random training samples for each of the possible fractions 10%, 20%, ..., 100% of all available labels. In Fig. 2 we can see that the growth/death rate based features as well as the PSimRank based features are not sensitive to training set size while the

XJaccard based ones are. That is, even though XJaccard is better in terms of performance than the other two feature sets considered it is more sensitive to the amount of training data used as well.

5.4 Temporal Content Ensembles

We build ensembles based on the temporal content features described in Section 3.2 and their combination themselves, with the static BM25 features, and with the content-based features of [59]. The performance comparison of temporal content-based ensembles is presented in Table 12.

Feature Set	**No. of Features**	**AUC**
Static BM25	10,000	0.736
Ave	10,000	0.749
AveDiff	10,000	0.737
Dev	10,000	0.767
DevDiff	10,000	0.752
Decay	10,000	0.709
Temporal combined	50,000	0.782
Temporal combined + BM25	60,000	**0.789**
Public content-based [59] + temporal	50,096	0.901
All combined	60,096	**0.902**

Table 12: Performance of ensembles built on temporal content-based features.

By combining all the content and link-based features, both temporal and static ones, we train an ensemble which incorporates all the previous classifiers. This combination resulted in an AUC of 0.908 meaning no significant improvement can be achieved with link-based features over the content-based ensemble.

6 Computational Resources

For the experiments we used a 45-node Hadoop cluster of dual core machines with 4GB RAM each as well as multi-core machines with over 40GB RAM. Over this architecture we were able to compute all features, some of which would require excessive resources either when used by a smaller archive or if the collection is larger or if fast classification is required for newly discovered sites during crawl time. Some of the most resource bound features involve the multi-step neighborhood in the page level graph that already requires approximation techniques for WEBSPAM-UK2007 [22].

We describe the computational requirements of the features by distinguishing update and batch processing. For batch processing an entire collection is analyzed at once, a procedure that is probably performed only for reasons of research. Update is probably the typical operation for a search engine. For an

Feature Set	**Step**	**Hours**	**Configuration**
Content (A) + BM25	Parsing	36	45 dual core Pentium-D 3.0GHz machines, 4GB RAM, Hadoop 0.21
	Feature generation	36	
	Selection of labeled pages	3	
Link	PageRank	10	5 eight-core Xeon 1.6GHz machines, 40+GB RAM
	Neighborhood	4	
	Local features	1	

Table 13: Processing times and cluster configurations for feature sets over ClueWeb09.

Internet Archive, update is also advantageous as long as it allows fast reaction to sample, classify and block spam from a yet unknown site.

6.1 Batch Processing

The first expensive step involves parsing to create terms and links. The time requirement scales linearly with the number of pages. Since apparently a few hundred page sample of each host suffices for feature generation, the running time is also linear in the number of hosts. For a very large collection such as ClueWeb09, distributed processing may be necessary. Over 45 dual core Pentium-D 3.0GHz machines running Hadoop 0.21, we parsed the uncompressed 9.5TB English part of ClueWeb09 in 36 hours. Additional tasks such as term counting, BM25 or content feature generation fits within the same time frame. If features are generated only a small labeled part of the data, it took us 3 hours to select the appropriate documents and additional processing time was negligible. Processing times are summarized in Table 13.

Host level aggregation allows us to proceed with a much smaller size data. However for aggregation we need to store a large number of partial feature values for all hosts unless we sort the entire collection by host, again by external memory or Map-Reduce sort.

After aggregation, host level features are inexpensive to compute. The following features however remain expensive:

- Page level PageRank. Note that this is required for all content features involving the maximum PageRank page of the host.
- Page level features involving multi-step neighborhood such as neighborhood size at distance k as well as graph similarity.

In order to be able to process graphs of ClueWeb09 scale (4.7 billion nodes and 17 billion edges), we implemented message passing C++ codes. Over a total 30 cores of six Xeon 1.6GHz machines, each with at least 40GB RAM, one PageRank and one Bit Propagation iteration both took approximately one hour while all other, local features completed within one hour.

Training the classifier for a few 100,000 sites can be completed within a day on a single CPU on a commodity machine with 4-16GB RAM; here costs

Configuration	**Number of Hosts**	**Feature Sets**	**Example**	**Expected Accuracy**	**Computation**
Small 1-2 machines	10,000	Content (A) BM25	subset of UK2007	0.80-0.87	Non-distributed
Medium 3-10 machines	100,000	Content (A) BM25, link	DC2010	0.87-0.90	MapReduce and Disk-based e.g. GraphChi
Large 10+ machines	1,000,000	Content (B) BM25, link	ClueWeb09	0.9+	MapReduce and Pregel

Table 14: Sample configurations for Web spam filtering in practice.

strongly depend on the classifier implementation. Our entire classifier ensemble for the labeled WEBSPAM-UK2007 hosts took a few hours to train.

6.2 Incremental Processing

As preprocessing and host level aggregation is linear in the number of hosts, this reduces to a small job for an update. This is especially true if we are able to split the update by sets of hosts; in this case we may even trivially parallelize the procedure.

The only nontrivial content based information is related to document frequencies: both the inverse document frequency term of BM25 [60] and the corpus precision and recall dictionaries may in theory be fully updated when new data is added. We may however approximate by the existing values under the assumption that a small update batch will not affect these values greatly. From time to time however all features beyond (Aa) need a global recomputation step.

The link structure is however nontrivial to update. While incremental algorithms exist to create the graph and to update PageRank type features [32, 33, 53], these algorithms are rather complex and their resource requirements are definitely beyond the scale of a small incremental data.

Incremental processing may have the assumption that no new labels are given, since labeling a few thousand hosts takes time comparable to batch process hundreds of thousands of them. Given the trained classifier, a new site can be classified in seconds right after its feature set is computed.

7 Conclusions

With the illustration over the 100,000 host WEBSPAM-UK2007, the half billion page ClueWeb09, and the 190,000 host DC2010 data sets, we have investigated the tradeoff between feature generation and spam classification accuracy. We observe that more features achieve better performance, however, when combining them with the public link based feature set we get only marginal performance gain. By using the WEBSPAM-UK2007 data along with seven previous monthly snapshots of the `.uk` domain, we have presented a survey of temporal features for Web spam classification. We investigated the performance of link, content and temporal[7] Web spam features with ensemble selection. As practical message, we may conclude that, as seen in Table 14, single machines may compute content and BM25 features for a few 10,000 hosts only. Link features need additional resources and either compressed, disk based or, in the largest configuration, Pregel-like distributed infrastructures.

We proposed graph similarity based temporal features which aim to capture the nature of linkage change of the neighborhoods of hosts. We have shown how to compute these features efficiently on large graphs using a Monte Carlo method. Our features achieve better performance than previously published methods, however, when combining them with the public link-based feature set we get only marginal performance gain.

By our experiments it has turned out that the appropriate choice of the machine learning techniques is probably more important than devising new complex features. We have managed to compile a minimal feature set that can be computed incrementally very quickly to allow to intercept spam at crawl time based on a sample of a new Web site. Sample configurations for Web spam filtering are summarized in Table 14.

Our results open the possibility for spam filtering practice in Internet archives who are mainly concerned about their resource waste and would require fast reacting filters. BM25 based models are suitable even for filtering at crawl time.

Some technologies remain open to be explored. For example, unlike expected, the ECML/PKDD Discovery Challenge 2010 participants did not deploy cross-lingual technologies for handling languages other than English. Some ideas worth exploring include the use of dictionaries to transfer a bag of words based model and the normalization of content features across languages to strengthen the language independence of the content features. The natural language processing based features were not used either, that may help in particular with the challenging quality attributes.

Acknowledgment

To Sebastiano Vigna, Paolo Boldi and Massimo Santini for providing us with the UbiCrawler crawls [10, 11]. In addition to them, also to Ilaria Bordino, Carlos Castillo and Debora Donato for discussions on the WEBSPAM-UK data sets [12].

To the large team of organizers and assessors for the complex labeling process of the DC2010 data set.

[7]The temporal feature data used in our research is available at: `https://datamining.sztaki.hu/en/download/web-spam-resources`

References

[1] J. Abernethy, O. Chapelle, and C. Castillo. WITCH: A New Approach to Web Spam Detection. In *Proceedings of the 4th International Workshop on Adversarial Information Retrieval on the Web (AIRWeb)*, 2008.

[2] L. D. Artem Sokolov, Tanguy Urvoy and O. Ricard. Madspam consortium at the ecml/pkdd discovery challenge 2010. In *Proceedings of the ECML/PKDD 2010 Discovery Challenge*, 2010.

[3] J. Attenberg and T. Suel. Cleaning search results using term distance features. In *Proceedings of the 4th international workshop on Adversarial information retrieval on the web*, pages 21–24. ACM New York, NY, USA, 2008.

[4] Z. Bar-Yossef, A. Z. Broder, R. Kumar, and A. Tomkins. Sic transit gloria telae: Towards an understanding of the web's decay. In *Proceedings of the 13th World Wide Web Conference (WWW)*, pages 328–337. ACM Press, 2004.

[5] Z. Bar-Yossef, I. Keidar, and U. Schonfeld. Do not crawl in the dust: different urls with similar text. *ACM Transactions on the Web (TWEB)*, 3(1):1–31, 2009.

[6] S. Barton. Mignify, a big data refinery built on hbase. In *HBASE CON*, 2012.

[7] L. Becchetti, C. Castillo, D. Donato, S. Leonardi, and R. Baeza-Yates. Link-based characterization and detection of web spam. In *Proceedings of the 2nd International Workshop on Adversarial Information Retrieval on the Web (AIRWeb)*, 2006.

[8] A. A. Benczúr, M. Erdélyi, J. Masanés, and D. Siklósi. Web spam challenge proposal for filtering in archives. In *AIRWeb '09: Proceedings of the 5th international workshop on Adversarial information retrieval on the web*. ACM Press, 2009.

[9] A. A. Benczúr, D. Siklósi, J. Szabó, I. Bíró, Z. Fekete, M. Kurucz, A. Pereszlényi, S. Rácz, and A. Szabó. Web spam: a survey with vision for the archivist. In *Proc. International Web Archiving Workshop*, 2008.

[10] P. Boldi, B. Codenotti, M. Santini, and S. Vigna. Ubicrawler: A scalable fully distributed web crawler. *Software: Practice & Experience*, 34(8):721–726, 2004.

[11] P. Boldi, M. Santini, and S. Vigna. A Large Time Aware Web Graph. *SIGIR Forum*, 42, 2008.

[12] I. Bordino, P. Boldi, D. Donato, M. Santini, and S. Vigna. Temporal evolution of the uk web. In *Workshop on Analysis of Dynamic Networks (ICDM-ADN'08)*, 2008.

[13] I. Bordino, D. Donato, and R. Baeza-Yates. Coniunge et impera: Multiple-graph mining for query-log analysis. In *Machine Learning and Knowledge Discovery in Databases*, pages 168–183. Springer, 2010.

[14] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[15] A. Z. Broder. On the Resemblance and Containment of Documents. In *Proceedings of the Compression and Complexity of Sequences (SEQUENCES'97)*, pages 21–29, 1997.

[16] R. Caruana, A. Munson, and A. Niculescu-Mizil. Getting the most out of ensemble selection. In *ICDM '06: Proceedings of the Sixth International Conference on Data Mining*, pages 828–833, Washington, DC, USA, 2006. IEEE Computer Society.

[17] R. Caruana, A. Niculescu-Mizil, G. Crew, and A. Ksikes. Ensemble selection from libraries of models. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, page 18, New York, NY, USA, 2004. ACM.

[18] C. Castillo, K. Chellapilla, and L. Denoyer. Web spam challenge 2008. In *Proceedings of the 4th International Workshop on Adversarial Information Retrieval on the Web (AIRWeb)*, 2008.

[19] C. Castillo and B. Davison. *Adversarial web search*, volume 4. Now Publishers Inc, 2011.

[20] C. Castillo, D. Donato, L. Becchetti, P. Boldi, S. Leonardi, M. Santini, and S. Vigna. A reference collection for web spam. *SIGIR Forum*, 40(2):11–24, December 2006.

[21] C. Castillo, D. Donato, A. Gionis, V. Murdock, and F. Silvestri. Know your neighbors: Web spam detection using the web topology. Technical report, DELIS – Dynamically Evolving, Large-Scale Information Systems, 2006.

[22] C. Castillo, D. Donato, A. Gionis, V. Murdock, and F. Silvestri. Know your neighbors: web spam detection using the web topology. *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 423–430, 2007.

[23] N. Chawla, N. Japkowicz, and A. Kotcz. Editorial: special issue on learning from imbalanced data sets. *ACM SIGKDD Explorations Newsletter*, 6(1):1–6, 2004.

[24] C. Chekuri, M. H. Goldwasser, P. Raghavan, and E. Upfal. Web search using automatic classification. In *Proceedings of the 6th International World Wide Web Conference (WWW)*, San Jose, USA, 1997.

[25] J. Cho and H. Garcia-Molina. The evolution of the web and implications for an incremental crawler. In *The VLDB Journal*, pages 200–209, 2000.

[26] J. Cho and H. Garcia-Molina. Synchronizing a database to improve freshness. In *Proceedings of the International Conference on Management of Data*, pages 117–128, 2000.

[27] E. Convey. Porn sneaks way back on web. *The Boston Herald*, May 1996.

[28] G. Cormack. Content-based Web Spam Detection. In *Proceedings of the 3rd International Workshop on Adversarial Information Retrieval on the Web (AIRWeb)*, 2007.

[29] G. Cormack, M. Smucker, and C. Clarke. Efficient and effective spam filtering and re-ranking for large web datasets. *Information retrieval*, 14(5):441–465, 2011.

[30] K. Csalogány, A. Benczúr, D. Siklósi, and L. Lukács. Semi-Supervised Learning: A Comparative Study for Web Spam and Telephone User Churn. In *Graph Labeling Workshop in conjunction with ECML/PKDD 2007*, 2007.

[31] N. Dai, B. D. Davison, and X. Qi. Looking into the past to better classify web spam. In *AIRWeb '09: Proceedings of the 5th international workshop on Adversarial information retrieval on the web.* ACM Press, 2009.

[32] P. Desikan, N. Pathak, J. Srivastava, and V. Kumar. Incremental page rank computation on evolving graphs. In *WWW '05: Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 1094–1095, New York, NY, USA, 2005. ACM.

[33] P. K. Desikan, N. Pathak, J. Srivastava, and V. Kumar. Divide and conquer approach for efficient pagerank computation. In *ICWE '06: Proceedings of the 6th international conference on Web engineering*, pages 233–240, New York, NY, USA, 2006. ACM.

[34] A. Dong, Y. Chang, Z. Zheng, G. Mishne, J. Bai, K. Buchner, R. Zhang, C. Liao, and F. Diaz. Towards recency ranking in web search. In *Proc. WSDM*, 2010.

[35] N. Eiron, K. S. McCurley, and J. A. Tomlin. Ranking the web frontier. In *Proceedings of the 13th International World Wide Web Conference (WWW)*, pages 309–318, New York, NY, USA, 2004. ACM Press.

[36] M. Erdélyi and A. A. Benczúr. Temporal analysis for web spam detection: An overview. In *1st International Temporal Web Analytics Workshop (TWAW) in conjunction with the 20th International World Wide Web Conference in Hyderabad, India.* CEUR Workshop Proceedings, 2011.

[37] M. Erdélyi, A. A. Benczúr, J. Masanés, and D. Siklósi. Web spam filtering in internet archives. In *AIRWeb '09: Proceedings of the 5th international workshop on Adversarial information retrieval on the web*. ACM Press, 2009.

[38] M. Erdélyi, A. Garzó, and A. A. Benczúr. Web spam classification: a few features worth more. In *Joint WICOW/AIRWeb Workshop on Web Quality (WebQuality 2011) In conjunction with the 20th International World Wide Web Conference in Hyderabad, India.* ACM Press, 2011.

[39] FastRandomForest. Re-implementation of the random forest classifier for the weka environment. `http://code.google.com/p/fast-random-forest/`.

[40] D. Fetterly and Z. Gyöngyi. Fifth international workshop on adversarial information retrieval on the web (AIRWeb 2009). 2009.

[41] D. Fogaras and B. Rácz. Scaling link-based similarity search. In *Proceedings of the 14th World Wide Web Conference (WWW)*, pages 641–650, Chiba, Japan, 2005.

[42] J. Fogarty, R. S. Baker, and S. E. Hudson. Case studies in the use of roc curve analysis for sensor-based estimates in human computer interaction. In *Proceedings of Graphics Interface 2005*, GI '05, pages 129–136, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 2005. Canadian Human-Computer Communications Society.

[43] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *Annals of statistics*, pages 337–374, 2000.

[44] G. Geng, X. Jin, and C. Wang. CASIA at WSC2008. In *Proceedings of the 4th International Workshop on Adversarial Information Retrieval on the Web (AIRWeb)*, 2008.

[45] X.-C. Z. Guang-Gang Geng, Xiao-Bo Jin and D. Zhang. Evaluating web content quality via multi-scale features. In *Proceedings of the ECML/PKDD 2010 Discovery Challenge*, 2010.

[46] Z. Gyöngyi and H. Garcia-Molina. Spam: It's not just for inboxes anymore. *IEEE Computer Magazine*, 38(10):28–34, October 2005.

[47] Z. Gyöngyi and H. Garcia-Molina. Web spam taxonomy. In *Proceedings of the 1st International Workshop on Adversarial Information Retrieval on the Web (AIRWeb)*, Chiba, Japan, 2005.

[48] Z. Gyöngyi, H. Garcia-Molina, and J. Pedersen. Combating web spam with TrustRank. In *Proceedings of the 30th International Conference on Very Large Data Bases (VLDB)*, pages 576–587, Toronto, Canada, 2004.

[49] M. R. Henzinger, R. Motwani, and C. Silverstein. Challenges in web search engines. *SIGIR Forum*, 36(2):11–22, 2002.

[50] A. Hotho, D. Benz, R. Jäschke, and B. Krause, editors. *Proceedings of the ECML/PKDD Discovery Challenge*. 2008.

[51] G. Jeh and J. Widom. SimRank: A measure of structural-context similarity. In *Proceedings of the 8th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 538–543, 2002.

[52] Y. joo Chung, M. Toyoda, and M. Kitsuregawa. A study of web spam evolution using a time series of web snapshots. In *AIRWeb '09: Proceedings of the 5th international workshop on Adversarial information retrieval on the web*. ACM Press, 2009.

[53] C. Kohlschütter, P. A. Chirita, and W. Nejdl. Efficient parallel computation of pagerank, 2007.

[54] Z. Kou and W. W. Cohen. Stacked graphical models for efficient inference in markov random fields. In *SDM 07*, 2007.

[55] Y. Lin, H. Sundaram, Y. Chi, J. Tatemura, and B. Tseng. Splog detection using content, time and link structures. In *2007 IEEE International Conference on Multimedia and Expo*, pages 2030–2033, 2007.

[56] T. Lynam, G. Cormack, and D. Cheriton. On-line spam filter fusion. *Proc. of the 29th international ACM SIGIR conference on Research and development in information retrieval*, pages 123–130, 2006.

[57] A. Niculescu-Mizil, C. Perlich, G. Swirszcz, V. Sindhwani, Y. Liu, P. Melville, D. Wang, J. Xiao, J. Hu, M. Singh, et al. Winning the KDD Cup Orange Challenge with Ensemble Selection. In *KDD Cup and Workshop in conjunction with KDD 2009*, 2009.

[58] V. Nikulin. Web-mining with wilcoxon-based feature selection, ensembling and multiple binary classifiers. In *Proceedings of the ECML/PKDD 2010 Discovery Challenge*, 2010.

[59] A. Ntoulas, M. Najork, M. Manasse, and D. Fetterly. Detecting spam web pages through content analysis. In *Proceedings of the 15th International World Wide Web Conference (WWW)*, pages 83–92, Edinburgh, Scotland, 2006.

[60] S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *In Proceedings of SIGIR'94*, pages 232–241. Springer-Verlag, 1994.

[61] G. Shen, B. Gao, T. Liu, G. Feng, S. Song, and H. Li. Detecting link spam using temporal information. In *ICDM'06.*, pages 1049–1053, 2006.

[62] D. Siklósi, B. Daróczy, and A. Benczúr. Content-based trust and bias classification via biclustering. In *Proceedings of the 2nd Joint WICOW/AIRWeb Workshop on Web Quality*, pages 41–47. ACM, 2012.

[63] A. Singhal. Challenges in running a commercial search engine. In *IBM Search and Collaboration Seminar 2004*. IBM Haifa Labs, 2004.

[64] S. Webb, J. Caverlee, and C. Pu. Predicting web spam with HTTP session information. In *Proceeding of the 17th ACM conference on Information and knowledge management*, pages 339–348. ACM, 2008.

[65] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques.* Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann, second edition, June 2005.

[66] B. Wu, V. Goel, and B. D. Davison. Topical TrustRank: Using topicality to combat web spam. In *Proceedings of the 15th International World Wide Web Conference (WWW)*, Edinburgh, Scotland, 2006.

[67] B. Zhou, J. Pei, and Z. Tang. A spamicity approach to web spam detection. In *Proceedings of the 2008 SIAM International Conference on Data Mining (SDM'08)*, pages 277–288. Citeseer, 2008.