Temporal Analysis for Web Spam Detection: An Overview*

Miklós Erdélyi[1,2] András A. Benczúr[1]
[1]Institute for Computer Science and Control, Hungarian Academy of Sciences
[2]University of Pannonia, Department of Computer Science & Systems Technology, Veszprém
{miklos, benczur}@ilab.sztaki.hu

ABSTRACT

In this paper we give a comprehensive overview of temporal features devised for Web spam detection providing measurements for different feature sets.

- We make a temporal feature research data set publicly available[1]. The features are based on eight UbiCrawler crawl snapshots of the `.uk` domain between October 2006 and May 2007 and use the WEBSPAM-UK2007 labels.
- We explore the performance of previously published temporal spam features and in particular 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.

Our experiments are conducted over the collection of eight `.uk` crawl snapshots that include WEBSPAM-UK2007.

Categories and Subject Descriptors

H.3 [**Information Systems**]: Information Storage and Retrieval; I.2 [**Computing Methodologies**]: Artificial Intelligence; I.7.5 [**Computing Methodologies**]: Document Capture—*Document analysis*

General Terms

Hyperlink Analysis, Feature Selection, Document Classification, Information Retrieval

Keywords

Web spam, Document Classification, Time series analysis, Hyperlink analysis

*This work was supported by the EU FP7 Project LIWA (Living Web Archives), LAWA (Large-Scale Longitudal Web Analytics) and by the grant OTKA NK 72845.

[1]`http://datamining.ilab.sztaki.hu/?q=en/downloads`

TWAW 2011, March 28, 2011, Hyderabad, India.

1. INTRODUCTION

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 [37, 28, 27]. Although recently there seems to be a slowdown in the achievements, temporal analysis appears as a new area with several recent papers [36, 31, 17, 30, 20].

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. We compare our result with the very strong baseline of the Web Spam Challenge 2008 data set.

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 [23].

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 [26] were achieved by using host level link features only, outperforming those who used all features [1].

The rest of this paper is organized as follows. After listing related results, in Section 2 we describe the features we add to the baseline set of [13] including new temporal features introduced first in this paper. In Section 3 we describe our classification framework. The results of the experiments to classify WEBSPAM-UK2007 by also relying on 7 additional crawl snapshots of the same domain can be found in Section 4.

1.1 Related Results

An excellent overview of Web spam filtering methods, both temporal and non-temporal approaches is found in [12]. When building our baseline classifier, we considered the known features as well as the classification methods used by the winners of the Web Spam Challenge 2008 [25] and the ECML/PKDD Discovery Challenge [26]. The baseline

ensemble classifier tested on two data sets is taken from our work [21].

Recently the evolution of the Web has attracted interest in defining features, signals for ranking [18] and spam filtering [36, 31, 17, 30, 20]. The earliest results investigate the changes of Web content with the primary interest of keeping a search engine index up-to-date [15, 16]. The decay of Web pages and links and its consequences on ranking are discussed in [3, 19]. One main goal of Boldi et al. [7] 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. [8].

Perhaps the first result on the applicability of temporal features for Web spam filtering is due to Shen et al. [36] 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 [23].

Another related result defines features based on the change of the content [17] who obtain page history from the Wayback Machine. They only present classification results for a selected subset of hosts and they do not compare their performance with the Web Spam Challenge 2008 results [11] as they only measure precision, recall and F-measure but not AUC (area under the ROC curve [24]). In order to be comparable with a larger set of spam detection techniques, we use the full 2,053 host Web Spam Challenge 2008 test set. In addition, we believe that AUC is 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.

In a preliminary result [20] we suggested the applicability of Jaccard and cosine similarity metrics for capturing content change of groups of Web pages. Compared to that result, in this paper we show full-scale results of applying term-weight based temporal content features. In addition, we derive features based on the multi-step linkage similarity of Web hosts. This set of features extends the growth and death rate of [36] that we use as baseline features.

For a broader outlook, temporal analysis is also applied for splog detection, i.e. manipulative blogs with the sole purpose to attract search engine traffic and promote affiliate sites. Lin et al. [31] consider the dynamics of self-similarity matrices of time, content and link attributes of posts. They use the Jaccard similarity, a technique that we are also applying in our experiments.

2. 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 [36] 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.

2.1 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 [36] 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 [14], and spam pages can be characterized by larger change of linkage when compared to honest pages [36].

In the following we review the features related to linkage growth and death from [36] in Section 2.1.1, 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 2.1.3 and Section 2.1.4, respectively.

2.1.1 Change Rate of In-links and Out-links

We compute the following features introduced by Shen et al. [36] 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):

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

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

- 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.

2.1.2 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.

We consider two graph similarity measures, XJaccard and PSimRank [23]; we also argue why SimRank [29] 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:

$$\begin{aligned} \mathrm{Sim}_{\ell+1}(u,v) &= 1, \text{ if } u = v; \\ \mathrm{Sim}_{\ell+1}(u,v) &= \sum_{\substack{v' \in I(v) \\ u' \in I(u)}} \mathrm{Sim}_{\ell}(u',v'). \end{aligned} \tag{1}$$

In order to apply SimRank for similarity of a node v between two snapshots t_0 and t_1, we apply (1) 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. Counterintuitively, 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.

2.1.3 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 [23]. 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

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

The XJac values can be approximated by the min-hash fingerprinting technique for Jaccard coefficients [10], as described in Algorithm 3 of [23]. 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 [23], 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 [36], 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 $\ell - 1$.

2.1.4 PSimRank Along Time

Next we define similarity over time based on PSimRank, a SimRank variant defined in [23] that can be applied similar to XJaccard in the previous section. As we saw in Section 2.1.2, 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 [23].

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 $\ell = 1 \ldots 4$. 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 [23], 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 $\mathrm{PSimRank}(w)$ of the neighbors w for each node v and derived features as for XJaccard.

2.2 Content and its Change

The content of Web pages can be deployed in content classification either via statistical features such as entropy [34] or via term weight vectors [39, 17]. Some of the more complex features that we do not consider in this work include language modeling [2].

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[2], remove stop words[3] 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 [35]:

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

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

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

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

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

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 $\text{tf}_{i,d}^{(T)} = 0$.

By using the term vectors as above, we calculate the temporal content features described in [17] in the following five groups.

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

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

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

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

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

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

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

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

- **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:

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

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

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

3. CLASSIFICATION FRAMEWORK

For the purposes of our experiments we computed all the public Web Spam Challenge content and link features of [13]. We applied the classification techniques found most effective in our work [21]. 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 [38].

The motivation for using ensemble selection is that recently this particular ensemble method gained more attention thanks to the winners of KDD Cup 2009 [33]. According to our experiments [21] ensemble selection performed significantly better than other classifier combination methods used for Web spam detection in the literature, such as log-odds based averaging [32] and bagging.

We used the ensemble selection implementation of Weka [38] for performing the experiments. The Weka implementation supports the proven strategies for avoiding 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.

3.1 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.

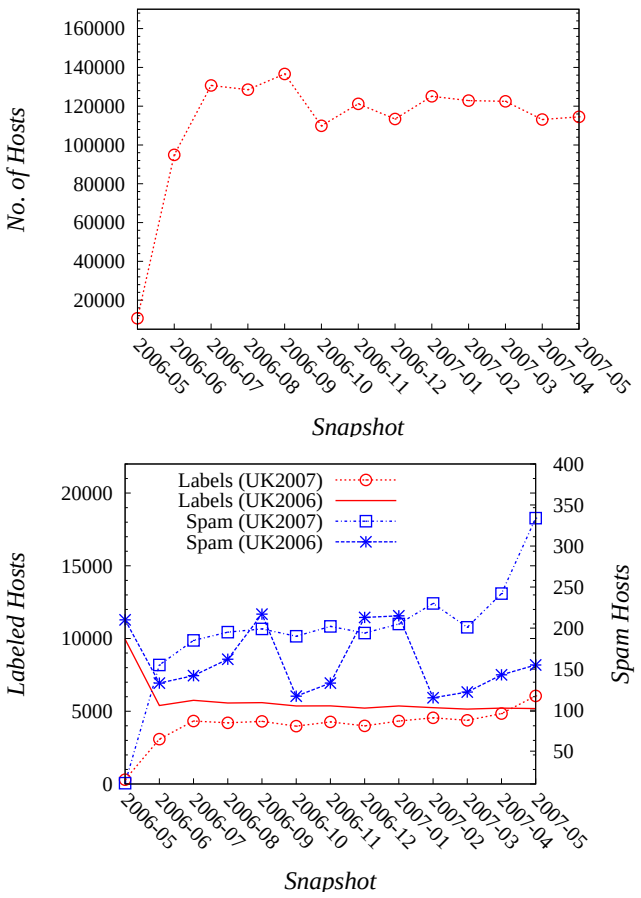

Figure 1: The number of total hosts as well as labeled hosts distinguished by the label set and the label value.

Random Forests: we use FastRandomForest [22] instead of the native Weka implementation for faster computation. The forests have 250 trees and, as suggested in [9], 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.

4. RESULTS AND DISCUSSION

Our data set is derived from the 13 `.uk` snapshots provided by the Laboratory for Web Algorithmics of the Università degli studi di Milano together with the Web Spam Challenge labels WEBSPAM-UK2007. 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. [8] who among others observe a relative low URL but high host overlap. 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 the last 8 snapshots contain a stable fraction of hosts (both labeled and unlabeled) for our experiments as seen in Fig. 1. From now on we restrict attention to the latter snapshots and the WEBSPAM-UK2007 labels only.

For calculating the temporal link-based features described in Section 2 we use the host level graph. Similar to the observation of [8], 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 crawl induced noise and problems with URL canonization [4] rule out the applicability of features based on the change of page-level linkage.

To make it easy to compare our results to previous results, we cite the Web Spam Challenge 2008 winner's performance in each table in the following, as published in their original paper [25]. 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 [25].

4.1 Classifier Models

In this subsection we describe the performance of various classifier ensemble combinations[4]. We do not aim to provide an exhaustive evaluation of all combinations. Instead, we concentrate our efforts on determining whether temporal information is valuable for Web spam detection.

For training and testing we use the official Web Spam Challenge 2008 training and test sets [13]. As it can be seen in Table 1 these show considerable class imbalance which makes the classification problem harder.

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

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

4.1.1 Temporal Link-only

First, we compare the temporal link features proposed in Section 2.1 with those published earlier [36]. Then, we build ensembles that combine the temporal with the public link-based features described by [5]. The results are summarized in Table 2.

Section	Feature Set	No. of Features	AUC
2.1.1	Growth/death rates	29	0.617
2.1.2	XJaccard + PSimRank	63	0.625
	Public link-based [5]	176	0.765
2.1.1	Public + growth/death rates	205	0.758
2.1.2	Public + XJaccard + PSimRank	239	**0.769**
	All link-based	268	0.765
	WSC 2008 Winner	-	0.852

Table 2: Performance of ensembles built on link-based features.

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

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, 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 2 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 [34]. The results for these experiments are shown in Table 3.

Section	Feature Set	No. of Features	AUC
2.1.1	Growth/death rates	29	0.605
2.1.2	XJaccard	42	0.626
2.1.3	PSimRank	21	0.593
2.1.2-3	XJaccard + PSimRank	63	0.610
	Public link-based [5]	176	**0.731**
2.1.1	Public + growth/death rates	205	0.696
2.1.2-3	Public + XJaccard + PSimRank	239	*0.710*
	All link-based	268	0.707
	WSC 2008 Winner	-	0.852

Table 3: Performance of bagged cost-sensitive decision trees trained on link-based features.

As it can be seen in Table 3, 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 3 and in Table 2 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 prevalent from Table 3, 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.

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.

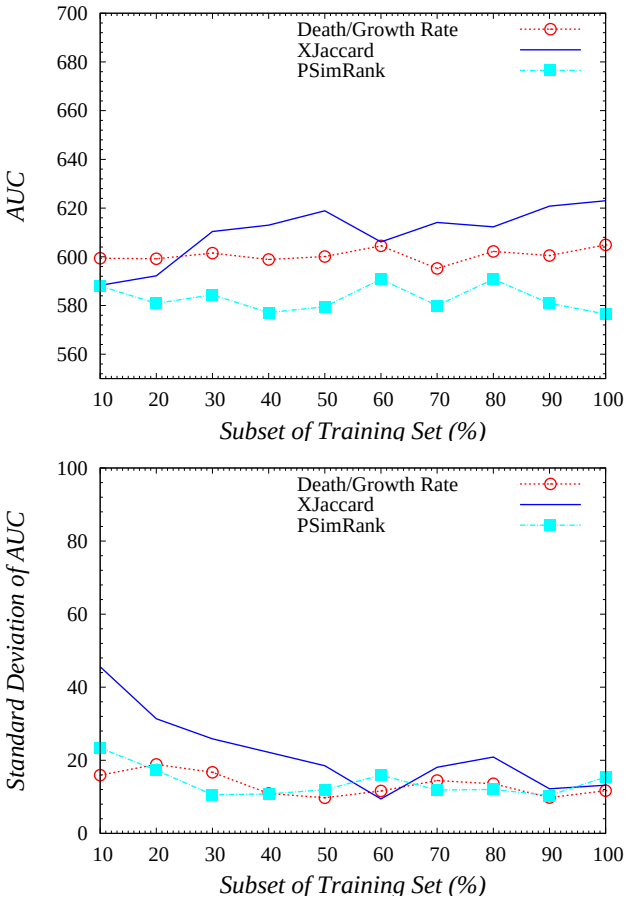

Figure 2: Sensitivity of temporal link-based features. Top: AUC values averaged across 10 measurements. Bottom: standard deviations of AUC for different training set sizes.

4.1.2 Content-only Ensemble

We build two ensembles, the first based on the Public content [34] features and the second on static term weight vector derived from the BM25 term weighting scheme (see Section 2.2). As seen in Table 4, the combination is by 5% stronger than the Web Spam Challenge 2008 winner [25].

Feature Set	No. of Features	AUC
Public content [34]	96	0.879
Public content + BM25	10096	**0.893**
WSC 2008 Winner [25]	-	0.852

Table 4: Performance of ensembles built on static content-based features.

4.1.3 Content-only Ensembles

We build ensembles based on the temporal content features described in Section 2.2 and their combination them-

selves, with the static BM25 features, and with the content-based features of [34]. The performance comparison of temporal content-based ensembles is presented in Table 5.

Feature Set	AUC
Static BM25	0.736
Ave	0.749
AveDiff	0.737
Dev	0.767
DevDiff	0.752
Decay	0.709
Temporal combined	0.782
Temporal combined + BM25	**0.789**
Public content-based [34] + temporal	0.901
All combined	**0.902**

Table 5: Performance of ensembles built on temporal content-based features.

4.1.4 Full Ensemble

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.

5. CONCLUSIONS

With the illustration over the 100,000 page WEBSPAM-UK2007 data along with 7 previous monthly snapshots of the `.uk` domain, we have presented a survey of temporal features for Web spam classification. We investigated the performance of both link- and content-based Web spam features with ensemble selection, focusing on temporal link-based features[5].

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. However, by using temporal information, we reach improvement in linkage based classification, a promising direction for filtering mixed language domains where content cannot be reliably used for classification [26].

Acknowledgment

To Sebastiano Vigna, Paolo Boldi and Massimo Santini for providing us with the UbiCrawler crawls [6, 7]. In addition to them, also to Ilaria Bordino, Carlos Castillo and Debora Donato for discussions on the WEBSPAM-UK data sets [8].

[5]The temporal feature data used in our research is available at: `http://datamining.ilab.sztaki.hu/?q=en/downloads`

6. REFERENCES

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

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

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

[8] 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.

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

[10] 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.

[11] 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.

[12] C. Castillo and B. D. Davison. Adversarial web search. *Foundations and Trends in Information Retrieval*, 4(5):377–486, 2010.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

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

[23] 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.

[24] 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.

[25] 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.

[26] 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.

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

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

[29] 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.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

[34] 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.

[35] 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.

[36] 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.

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

[38] 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.

[39] 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.