# Link Based Small Sample Learning for Web Spam Detection

Guang-Gang Geng
Computer Network Information Center
Chinese Academy of Sciences
Beijing 100190, P. R. China
gengguanggang@cnnic.cn

Qiu-Dan Li
Institute of Automation
Chinese Academy of Sciences
Beijing 100190, P. R. China
qiudan.li@ia.ac.cn

Xin-Chang Zhang
Computer Network Information Center
Chinese Academy of Sciences
Beijing 100190, P. R. China
zhangxinchang@cnnic.cn

## ABSTRACT

Robust statistical learning based web spam detection system often requires large amounts of labeled training data. However, labeled samples are more difficult, expensive and time consuming to obtain than unlabeled ones. This paper proposed link based semi-supervised learning algorithms to boost the performance of a classifier, which integrates the traditional Self-training with the topological dependency based link learning. The experiments with a few labeled samples on standard WEBSPAM-UK2006 benchmark showed that the algorithms are effective.

## Categories and Subject Descriptors

H.5.4 [**Information Interfaces and Presentation**]: Hypertext/Hypermedia; K.4.m [**Computer and Society**]: Miscellaneous; H.4.m [**Information Systems**]: Miscellaneous

## General Terms

Measurement, Experimentation, Algorithms

## Keywords

Link spam, Content spam, Web spam, Machine learning

## 1. INTRODUCTION

Statistical learning based web spam detection has demonstrated its superiority for being easy to adapt to newly developed spam techniques[1][3][5]. The robust machine learning based web spam detection system requires large amounts of labeled training samples. However, labeled samples are difficult, expensive and time consuming to obtain, as they require the efforts of experienced human annotators. The scarcity of training data becomes one of the major challenges for web spam detection. Meanwhile unlabeled data may be relatively easy to collect.

Semi-supervised learning addresses this problem by using large amounts of unlabeled data, together with the labeled data, to build better classifiers. Self-training is a commonly used technique for semi-supervised learning[4]. However the learning ability of the Self-training is limited. Based on the fact that the neighboring nodes are often having the same properties[1][3], we proposed link-based bootstrapping learning algorithms—Link-training and LS-training, which make full use of the self-learning of classifiers and the topological dependency of web nodes. Experiments on WEBSPAM-UK2006 benchmark show that the Link-Training and LS-training algorithm are effective.


*WWW 2009*, April 20–24, 2009, Madrid, Spain.
ACM 978-1-60558-487-4/09/04.

## 2. LINK BASED LEARNING ALGORITHM

Link-training algorithm is based on the Self-training and link learning, which makes full use of the classifier's self-learning ability and the topological dependency on the Web graph. Figure 1 is the flow chart of Link-training algorithm[1].

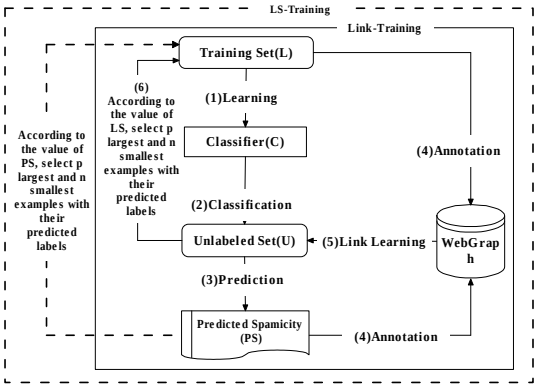


**Figure 1: Flow Chart of Link based Learning Algorithm.**

In Link-training, a classifier is first trained with a small labeled data set. The trained classifier is then used to classify the unlabeled data and give them a predicted spamicity($PS$) value ($PS$ is computed with Formula 1, where $P_{spam}(x)$ represents the probability of $x$ belonging to spam.). Then annotate the WebGraph with the $PS$ values of samples in $U$ and $L$ (the $PS$ values of spam and non-spam samples in $L$ are 1 and 0 respectively). In link learning step, the link spamicity($LS$) of unlabeled samples will be computed according to their neighbors. Based on value of $LS$, $p$ largest samples and $n$ smallest samples are converted into labeled ones with their predicted labels. The training procedure repeats until the number of iteration reaches a specified value.

$$PS(x) = \frac{P_{spam}(x)}{P_{spam}(x) + P_{normal}(x)} \quad (1)$$

[1]The dotted line on the left is a part of LS-Training, not included in Link-Training.

Inspired by the effectiveness of the re-extracted neighbor features in our previous work[3], we compute the link spamicity($LS$) as follows:

$$LS(h) = \frac{\sum_{v \in N(h)}(PS(v) \times weight(h,v))}{\sum_{v \in N(h)} weight(h,v)} \quad (2)$$

where $v, h$ are the hosts, $weight(h,v)$ is the weight of host $h$ and $v$, $weight(h,v) \in \{1, n, log(n)\}$, where $n$ is the number of hyperlinks between $h$ and $v$. $N(h) \in inlink(h) \bigcup outlink(h)$ $inlink(h)$ represents the inlink set of $h$, and $outlink(h)$ is the outlink set of $h$.

Algorithm 1 is the detailed description of the Link-training.

**Algorithm 1** Link-Training Algorithm

*Input* : $L$: Labeled training set
$U$: Unlabeled examples set
$T$: Test set
$C$: Classifier
$K$: Iterations
$G$: Host level hyperlink graph
$n, p$: The number of selected non-spam and spam samples in each iteration

1: $i = 0$
2: **while** $i < K$ **do**
3: Train classifier $C$ with $L$
4: Detect the examples in $U$ with $C$, compute their $PS$ values with Formula 1
5: Annotate $G$ with the $PS$ values of samples in $U$ and $L$ (the $PS$ value of spam and non-spam samples in $L$ are 1 and 0 respectively)
6: Perform link learning on annotated $G$; Compute the $LS$ values with Formula 2
7: According to the values of $LS$, select $p$ largest and $n$ smallest examples as spam and normal respectively, put the $n + p$ samples to $L$, and delete them from $U$ (The principle of choosing $p$ and $n$ is to keep the ratio of non-spam and spam unchanged in training set.)
8: $i = i + 1$
9: **end while**
10: Train classifier $C$ on the train set $L$
11: Test the samples in $T$ with the trained classifier $C$

*Output* : Web spam detection result on test set $T$

Based on Link-training and Self-training, we further propose LS-training algorithm. Compared with Link-training, LS-training also injects the samples with maximal and minimal values of $PS$ to $L$ in every iterations, which is descripted with dot line in Figure 1. LS-training algorithm boosts the labeled set from two different perspectives, which is similar to the starting point of Co-training.

# 3. EXPERIMENTS

## 3.1 Data Collection

WEBSPAM-UK2006 [2] is used in our experiments. The collection includes 77.9 million pages, corresponding to roughly 11400 hosts. We use all the labeled data with their homepage in the summarized samples.

## 3.2 Features and Classifier

The features extracted for learning include 72 content features and 81 link features (41 page-level link features and 40 host-level link features), which are the same as the features we used in Web Spam Challenge 2008[5]. The detection algorithm employed in the experiment is adaboost, and the weak classifier for adaboost is stump.

## 3.3 Experiment Results

For the UK2006, we randomly select 25% data are kept as test samples. From the remaining 75% data, we randomly select 100 samples as labeled set, while the rest samples are used as unlabeled set. The experiment results is obtained with 20 times random partition. The iterations $K$ is set to 50 and boolean weight is used. For keeping the ratio of spam and normal samples unchanged, $n$ is set to 15, and $p$ is set to 6. In other words, for LS-training 42 unlabeled samples with the predicted labels are put into the labeled set in each iteration. AUC and F1-measure are used to measure the performance.

Figure. 2 describes the performance of web spam detection with LS-training, Link-training and Self-training. From the figure, we can see that LS-training and Link-training are more effective than Self-training algorithm. With the increase in the number of iterations, both F1-measure and AUC have a significant improvement, while the performance of Self-training was poor.

The LS-training performance after 50 iterations with 100 labeled samples is F1-measure= 0.8037 and AUC=0.9385. Compared with the best results achieved on the whole data set[1][3], the results are inspiring.

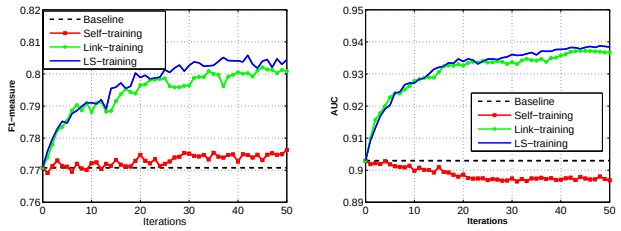


**Figure 2: Comparison of web spam detection performance with different algorithm**

# 4. CONCLUSION

In this paper, we proposed two link-based semi-supervised learning algorithms to detect web spam on small labeled samples set. Experiments show that the algorithms can boost the performance of the classifier effectively, which gives a feasible scheme for small samples detection. Another potential application of the proposed algorithms is to assist the human with annotating the large-scale unlabeled Web nodes preliminarily.

# 5. REFERENCES

[1] C. Castillo, D. Donato, A. Gionis, V. Murdock, and F. Silvestri. Know Your Neighbors: Web Spam Detection Using the Web Topology. *SIGIR'07*, May, 2007.
[2] Yahoo! Research: Web Collection UK-2006. Crawled by University of Milan, 2007.
[3] G.G.Geng, C.H.Wang, Q.D.Li. Improving web spam detection with re-extracted features, *WWW'08*, 2008.
[4] M Mohri, B Roark. Effective selftraining for parsing, *Proceedings of HLT-NAACL'06*, America, 2006
[5] G.G.Geng, X.B.Jin,C.H.Wang. CASIA at WSC2008, *Web Spam Challenge'08* http://webspam.lip6.fr,2008.