Transductive Link Spam Detection

Dengyong Zhou
Microsoft Research
One Microsoft Way
Redmond, WA 98052
denzho@microsoft.com

Christopher J.C. Burges
Microsoft Research
One Microsoft Way
Redmond, WA 98052
cburges@microsoft.com

Tao Tao
Microsoft Corp.
One Microsoft Way
Redmond, WA 98052
taotao@microsoft.com

ABSTRACT

Web spam can significantly deteriorate the quality of search engines. Early web spamming techniques mainly manipulate page content. Since linkage information is widely used in web search, link-based spamming has also developed. So far, many techniques have been proposed to detect link spam. Those approaches are basically built on link-based web ranking methods.

In contrast, we cast the link spam detection problem into a machine learning problem of classification on directed graphs. We develop discrete analysis on directed graphs, and construct a discrete analogue of classical regularization theory via discrete analysis. A classification algorithm for directed graphs is then derived from the discrete regularization. We have applied the approach to real-world link spam detection problems, and encouraging results have been obtained.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Retrieval models; I.2.6 [**Learning**]: Concept learning; G.2.2 [**Graph Theory**]: Graph labeling

General Terms

Theory, algorithms, performance

Keywords

Directed graph, discrete analysis, discrete regularization, transductive inference, link spam

1. INTRODUCTION

Web search engines are playing an increasingly important role in our daily life. In order to attract more traffic, the authors of some websites try to manipulate the ranking lists from web search engines such that their websites are ranked at top positions. The manipulation techniques depends on how a web search engine ranks the web pages for a given query. Such techniques are called spamming.

The first generation of web search engines are mainly built on the classical vector space model of information retrieval [11]. Under this model, if a query term repeatedly occurs in a web page, then the web page will be given a high rank for that query. Thus web spam pioneers manipulated the content web pages through keyword stuffing [21]. Specifically, they added many copies of highly searched keywords to their web pages in an attempt to gain traffic. Those stuffed keywords may be hidden from users by using a variety of mechanisms such as white text on a white background.

Since link analysis algorithms were incorporated into search engines [19, 4, 16], corresponding spamming techniques have been developed [13, 29, 22]. In those algorithms, the more links that point to a web page, the more important the web page may seem to a search engine. Thus web spammers will try to create a large number of links to their web pages by getting unrelated web sites to link to them, using automated techniques to post links to their web sites onto other web pages or just creating lots of their own pages and web sites and linking them all together. In addition, some web sites reciprocally exchange links. This is likely to be spam if those web sites are unrelated.

Since web spam can mislead a search engine to return low quality or even entirely irrelevant information to users, we need to remove the spam pages from the web corpus accessed by the search engine. Human experts generally can effectively identify web spam. However, it is quite easy for a spammer to create a large number of spam pages and to manipulate their link structure. So it is impractical to detect web spam only using human judges, and automated methods are needed. The automatic approach can be supervised (in which some spam examples are provided) or unsupervised (in which they are not).

In this paper, we cast the link spam detection problem into a machine learning problem of classification on directed graphs. For the latter, we have a directed graph, and some nodes on the graph that have been labeled, for instance, as spam or normal. The task is to label the remaining unclassified nodes. For attacking this problem, we develop discrete analysis on directed graphs, and then use the developed discrete operators to establish discrete regularization theory. As is well known, many learning approaches, like Support Vector Machines (SVMs) can be understood under regularization framework [27, 28]. Similarly, our classification approach will be derived from our discrete analog of regularization.

Classification on directed graphs belongs to transductive inference [27] rather than induction. Given a set of objects

AIRWeb '07, May 8, 2007 Banff, Alberta, Canada.
Copyright 2007 ACM 978-1-59593-732-2 ...$5.00.

$X = \{x_1, x_2, \ldots, x_l, x_{l+1}, \ldots, x_n\}$ from a domain of $\mathcal{X}$ (e.g., $\mathbb{R}^d$) of which the first l objects are labeled as $y_1, \ldots, y_l \in \mathcal{Y} = \{1, -1\}$, the goal is to predict the labels of the remaining unlabeled objects indexed from $l + 1$ to n. Any supervised learning algorithm can be applied to the above inference problem, e.g., by training a classifier $f : \mathcal{X} \to \mathcal{Y}$ with the set of pairs $\{(x_1, y_1), \ldots, (x_l, y_l)\}$, and then using the trained classifier f to predict the labels of the unlabeled objects. Following this methodology, one will have to estimate a classification function defined on the whole domain $\mathcal{X}$ before predicting the labels of the unlabeled objects. Estimating a classification function defined on the whole domain $\mathcal{X}$ is more complex than the original problem which only requires predicting the labels of the given unlabeled objects, and a better approach is to directly predict the labels of the given unlabeled objects. In other words, we should estimate a discrete classification function which is defined on the given objects in X only. Such an estimation problem is called transductive inference. In psychology, transductive reasoning means linking particular to particular with no consideration of the general principles. It is generally used by very young children. In contrast, deductive reasoning, which is used by adults and older children, means the ability to come to a specific conclusion based on a general premise.

This paper is organized as follows. In Section 2, we review the link spam detection literature. We then present some basic notions on directed graphs and Markov chains in Section 3, as a primer and to establish notation. In Section 4, we define a number of discrete operators for directed graphs, including gradient, divergence, and Laplacian. In particular, those discrete operators and also their properties are constructed mainly in a coordinate-free fashion, as is sometimes used in theoretical physics. In Section 5, we develop a discrete analog of classical regularization theory. This analog is defined on directed graphs, and it leads to a new classification algorithm. Finally, we validate our regularization based approach on real-world link spam detection, and the experimental results are shown in Section 5.

2. RELATED WORK

Brin and Page proposed a ranking measure on web pages from a model of a web surfer randomly following hyperlinks and this rank measure is well known as PageRank [4]. Specifically, at each web page, the web surfer either selects an outlink uniformly at random to follow with a certain probability, or jumps to a new web page selected from the whole web uniformly at random again with the remaining probability. The stationary probability of a web page in this random walk is regarded as the ranking score of the web page. The basic assumption behind PageRank is that a hyperlink from one page to another is a recommendation of the second page by the author of the first. If we use this assumption recursively, then a web page is considered to be important if many important web pages point to it.

In Brin and Page's rank measure, the use of random jumps to uniformly selected pages is a way to deal with the problem that some high quality web pages have no outlinks although they are pointed by many other web pages. However, in addition solving this problem, random jumps turn out to be a powerful parameter to adjust page ranks. In particular, if we let the random web surfer in the random process jump to a certain part of the web, that is of particular interest to a user, much more frequently than other web pages, the resulted rank is then personalized to that user [4, 14].

Random jumps can also be adopted to combat web spam, and the corresponding algorithm is called TrustRank [13]. The basic idea is to let the random web surfer jump to a set of pages which have been judged as of high quality by human experts. With this choice for the random jumps, the stationary probability of a web page is regarded as its trust score, and a web page with a trust score smaller than a chosen threshold value will be considered to be spam. TrustRank can also be understood as follows: initially, only the selected good seed pages have trust scores equal to 1, and the trust scores of other web pages are 0; each seed page then iteratively propagates its trust scores to its neighbors, and its neighbors further propagate their received scores to their neighbors. The basic assumption underlying this algorithm is that web pages of high quality seldom point to spam ones.

A counterpart of TrustRank has been developed, and it is called AntiTrust [22]. In this approach, the random web surfer either selects an inlink uniformly at random to reversely follow with a certain probability, or jumps to a new web page randomly selected from a web page set which have been judged as spam by human experts with the remaining probability. The stationary probability of a web page is referred to as its AntiTrust scores. A web page will be classified as spam if its score larger than a chosen threshold value. In terms of propagation, the scores in AntiTrust are propagated in the reverse direction along the inlinks, while the scores in TrustRank are propagated along the outlinks. The basic assumption underlying AntiTrust is that a web page pointing to spam pages is likely to be spam. The experimental results in [22] shows that AntiTrust outperforms TrustRank at the task of detecting web spam with high precision at various levels of recall.

Baeza-Yates, Boldi and Castillo propose a variant of PageRank, which is called functional ranking [2]. Essentially, they consider a general ranking function that depends on incoming paths of varying lengths weighted by some chosen damping function that decreases with distance. PageRank then can be understood as a rank function with a particular dumping function shown to be $(1-\alpha)\alpha^t$, where α is a damping factor in $]0, 1[$, typically 0.85 [4], and t is the length of a path. The functional ranking can be expected to be useful in link spam detection. The web page participating in a link farm can gain a high PageRank score because it has many inlinks from its neighbors. Hence, we can demote the web page via choosing a damping function that ignores the direct contribution of the first level of links.

The link spam detection issue can also be considered in a typical machine learning fashion [21, 6]. In this methodology, first we need to design some features that are useful in detecting spam, and represent each web page as a vector of which each element describe one kind of spam feature. Those features can be the number of inlinks, the number of outlinks, the PageRank score, and so on. We then choose a well studied classifier, like a neural network, decision tree or SVM, and train it with a set of examples of normal and spam web pages which have been judged by human experts. Finally, the trained classifier is used to predict if a given web page is spam or not. The main issue in this methodology is that the efficiency of a spam feature is generally validated only on the web pages which are not sampled from the entire web uniformly at random, but from large web sites and

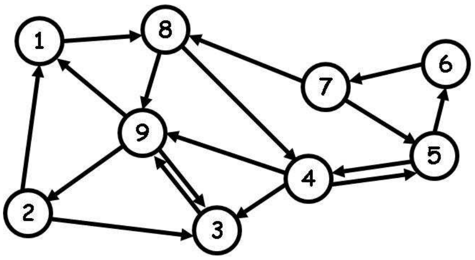

Figure 1: A directed graph with 9 nodes and 17 edges. Note that this graph is strongly connected.

highly ranked web pages. Consequently, the trained classifier is biased to those selected web pages, and it does not generalize to the whole web effectively.

3. BASIC NOTIONS

Let $G = (V, E)$ denote a directed graph, where V is the set of vertices, and E the set of edges. For a given edge $e \in E$, denote the initial vertex of e by e^-, and the terminal vertex of e by e^+. We also denote by (u, v) an edge from the vertex u to the vertex v. It is clear that an undirected graph can be regarded as a directed graph with each edge being double oriented. A graph G is weighted if it is associated with a function $w : E \to \mathbb{R}^+$ which assigns a positive number $w(e)$ to each edge e of G. Let $G = (V, E, w)$ denote a weighted directed graph. The function w is called the weight function of G. The in-degree d^- and the out-degree d^+ of a vertex $v \in V$ are respectively defined as

$$d^-(v) = \sum_{\{e|e^+=v\}} w(e), \text{ and } d^+(v) = \sum_{\{e|e^-=v\}} w(e). \quad (1)$$

A path is a tuple of vertices $(v_1, v_2, \ldots, v_p)$ with the property that $(v_i, v_{i+1}) \in E$ for $1 \le i \le p-1$. We say that a directed graph is strongly connected when for every pair of vertices u and v there is a path in which $v_1 = u$ and $v_p = v$. For a strongly connected graph, there is an integer $k \ge 1$ and a unique partition $V = V_0 \cup V_1 \cup \cdots \cup V_{k-1}$ such that for all $0 \le r \le k-1$ each edge $(u, v) \in E$ with $u \in V_r$ has $v \in V_{r+1}$, where $V_k = V_0$, and k is maximal, that is, there is no other such partition $V = V_0^{'} \cup \cdots \cup V_{k^{'}-1}^{'}$ with $k^{'} > k$. When $k = 1$, we say that the graph is aperiodic; otherwise we say that the graph is periodic.

For a given weighted directed graph, there is a natural random walk on the graph with the transition probability function $p : V \times V \to \mathbb{R}^+$ defined by

$$p(u, v) = \frac{w(u, v)}{d^+(u)} \quad (2)$$

for all $(u, v) \in E$, and 0 otherwise. If the graph is strongly connected, there is a unique function $\pi : V \to \mathbb{R}^+$ which satisfies

$$\sum_{u \in V} \pi(u)p(u, v) = \pi(v), \text{ and } \sum_{v} \pi(v) = 1. \quad (3)$$

The first equation in (3) is called the balance equation, and π is called the Perron vector. For a general directed graph, there is no closed form solution for π (see [8]). If the graph is both strongly connected and aperiodic, the random work defined by Equation (2) converges to the Perron vector π. Unless stated otherwise, the directed graphs we considered are always assumed to be strongly connected (Figure 1).

4. ANALYSIS ON DIRECTED GRAPHS

We define discrete operators on directed graphs, which are slight variants of the definitions presented in [31]. Those operators are discrete analogs of the corresponding differential operators on Riemannian manifolds, for example, see [18]. The discrete operators are then used to develop a discrete analog of classical regularization theory [26, 28]. Consequently, as in other regularization based machine learning algorithms in vectorial spaces, for instance, SVMs, our classification algorithm for directed graphs will be naturally derived from the discrete regularization.

Let $\mathcal{F}(V)$ denote the set of all real-valued functions on V, and $\mathcal{F}(E)$ the set of all real-valued functions on E. The function set $\mathcal{F}(V)$ can be regarded as a Hilbert space $\mathcal{H}(V)$ with the inner product defined by

$$\langle \varphi, \phi \rangle_{\mathcal{H}(V)} = \sum_{v \in V} \varphi(v)\phi(v)\pi(v), \quad (4)$$

where $\varphi, \phi \in \mathcal{F}(V)$. Let $c(e) = \pi(e^-)p(e)$. The number $c(e)$ is called the *ergodic flow* on e. It is easy to check that the ergodic flow is a *circulation*, that is,

$$\sum_{\{e|e^-=v\}} c(e) = \sum_{\{e|e^+=v\}} c(e), \ \forall v \in V. \quad (5)$$

A Hilbert space $\mathcal{H}(E)$ over $\mathcal{F}(E)$ can be constructed with the inner product defined by

$$\langle \vartheta, \psi \rangle_{\mathcal{H}(E)} = \sum_{e \in E} \vartheta(e)\psi(e)c(e), \quad (6)$$

where $\vartheta, \psi \in \mathcal{F}(E)$.

We define the *discrete gradient* $\nabla : \mathcal{H}(V) \to \mathcal{H}(E)$ as an operator

$$(\nabla\varphi)(e) := \varphi(e^+) - \varphi(e^-), \forall \varphi \in \mathcal{H}(V). \quad (7)$$

For simplicity, $(\nabla\varphi)(e)$ will be also denoted as $\nabla_e\varphi$. For gaining an intuition of this definition, one may imagine a set of buckets, and some of them are connected by tubes. Assume a tube e which connects buckets e^- and e^+, and the quantities of fluid in buckets e^- and e^+ to be $\varphi(e^-)$ and $\varphi(e^+)$. Then the flow though the tube should be proportional to the pressure difference and hence to $\varphi(e^+) - \varphi(e^-)$. When the fluid distributes itself uniformly among buckets, that is, φ is constant, the pressure differences will disappear and consequently there will be no flow in tubes any more, that is, $\nabla\varphi$ vanishes everywhere.

As in the continuous case, we define the *discrete divergence* $\operatorname{div} : \mathcal{H}(E) \to \mathcal{H}(V)$ as the dual of $-\nabla$, that is,

$$\langle \nabla\varphi, \psi \rangle_{\mathcal{H}(E)} = \langle \varphi, -\operatorname{div}\psi \rangle_{\mathcal{H}(V)}, \quad (8)$$

where $\varphi \in \mathcal{H}(V), \psi \in \mathcal{H}(E)$. By a straightforward computation, we obtain

$$(\operatorname{div}\psi)(v) = \frac{1}{\pi(v)} \left(\sum_{\{e|e^-=v\}} c(e)\psi(e) - \sum_{\{e|e^+=v\}} c(e)\psi(e) \right). \quad (9)$$

By following the above fluid model, the divergence measures the net flows at buckets. Now we can generalize the concept of circulation in terms of divergence. A function $\psi \in \mathcal{H}(E)$ is called a circulation if and only if $\operatorname{div}\psi = 0$.

We define the *discrete Laplacian* $\Delta : \mathcal{H}(V) \to \mathcal{H}(V)$ by

$$\Delta := -\frac{1}{2}\operatorname{div}\circ\nabla. \tag{10}$$

Compared with its counterpart in the continuous case, the additional factor in Equation (10) is due to edges being oriented. From Equation (10), we have

$$\langle \Delta\varphi, \phi\rangle_{\mathcal{H}(V)} = \frac{1}{2}\langle \nabla\varphi, \nabla\phi\rangle_{\mathcal{H}(E)} = \langle \varphi, \Delta\phi\rangle_{\mathcal{H}(V)} \tag{11}$$

Note that the first equation in (11) is a discrete analog of Green's formula. In addition, Equations (11) imply that Δ is self-adjoint. In particular, when $\varphi = \phi$, we have

$$\langle \Delta\varphi, \varphi\rangle_{\mathcal{H}(V)} = \frac{1}{2}\langle \nabla\varphi, \nabla\varphi\rangle_{\mathcal{H}(E)} = \frac{1}{2}\|\nabla\varphi\|^2_{\mathcal{H}(E)}, \tag{12}$$

which implies that Δ is positive semi-definite. By substituting Equations (7) and (9) into Equation (10), we have

$$(\Delta\varphi)(v) = \varphi(v) - \frac{1}{2\pi(v)} \cdot \left(\sum_{\{e|e^+=v\}} c(e)\varphi(e^-) + \sum_{\{e|e^-=v\}} c(e)\varphi(e^+) \right) \tag{13}$$

When the graph is undirected, that is, each edge being double oriented, Equation (13) reduces to

$$(\Delta\varphi)(v) = \varphi(v) - \frac{1}{d(v)}\sum_{u\sim v} w(u,v)\varphi(v). \tag{14}$$

Equation (14) has been widely used to define the Laplacian for an undirected graph, for example, see [9]. For the connection between the undirected Laplacian and the Laplacian in the continuous case, we refer the reader to [15]. Define a family of functions $\{\delta_v\}_{v\in V}$ with $\delta_v(u) = \mathbb{I}_{u=v}$, which is clearly a basis of $\mathcal{H}(V)$. The matrix form of Δ with respect to this basis has the following components:

$$\Delta_{\mathrm{am}}(u,v) = \begin{cases} -\dfrac{c(u,v)+c(v,u)}{2\pi(u)} & u \neq v, \\ 1 & u = v. \end{cases} \tag{15}$$

This matrix is not symmetric. However, if we choose another basis $\{\pi^{-1/2}(v)\delta_v\}_{v\in V}$, then we can represent Δ as a symmetric matrix

$$\Delta_{\mathrm{sm}}(u,v) = \begin{cases} -\dfrac{c(u,v)+c(v,u)}{2\sqrt{(\pi(u)\pi(v))}} & u \neq v, \\ 1 & u = v. \end{cases} \tag{16}$$

This matrix has been used to define the Laplacian for a directed graph [8, 31].

5. LEARNING ON DIRECTED GRAPHS

Given a directed graph $G = (V, E, w)$, and a discrete label set $L = \{-1, 1\}$, the vertices in a subset $S \subset V$ have labels in L. The task is to predict the labels of those unclassified vertices in S^c, the complement of S. The link spam detection problem can be cast into classification on a directed graph (Figure 2).

Define a function y with $y(v) = 1$ or -1 if $v \in S$, and 0 if $v \in S^c$. For classifying those unclassified vertices in S^c, we define a discrete regularization

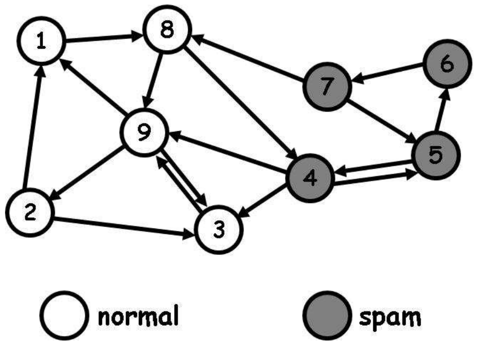

Figure 2: Classification on a web graph. Those nodes are labeled as two classes, normal and spam web pages.

$$\operatorname*{argmin}_{\varphi\in\mathcal{H}(V)} \left\{ \|\nabla\varphi\|^2_{\mathcal{H}(E)} + C\|\varphi - y\|^2_{\mathcal{H}(V)} \right\}, \tag{17}$$

where $C > 0$ is the regularization parameter. Intuitively, in the objective function, the first term forces the classification function to be as smooth as possible, and the second term forces the classification function to fit the given labels as well as possible.

When choosing the basis $\{\delta_v\}_{v\in V}$, Equation (17) can be written as

$$\operatorname*{argmin}_{\varphi\in\mathcal{H}(V)} \left\{ \sum_{e\in E} \pi(e^-)p(e)\left(\varphi(e^+) - \varphi(e^-)\right)^2 + C\sum_{v\in V}\pi(v)\left(\varphi(v) - y(v)\right)^2 \right\}. \tag{18}$$

If we scale each function in $\mathcal{H}(V)$ with a factor $\pi^{-1/2}$ (in other words, choose another basis $\{\pi^{-1/2}(v)\delta_v\}_{v\in V},$), then Equation (18) will be transformed into

$$\operatorname*{argmin}_{\phi\in\mathcal{H}(V)} \left\{ \sum_{e\in E} \pi(e^-)p(e)\left(\frac{\varphi(e^+)}{\sqrt{\pi(e^+)}} - \frac{\varphi(e^-)}{\sqrt{\pi(e^-)}} \right)^2 + C\sum_{v\in V}\left(\varphi(v) - y(v)\right)^2 \right\}. \tag{19}$$

This is the classification approach proposed in [31]. However, Equation (18) looks much more natural than Equation (19).

We have many choices in defining a random walk over a given directed graph. Here we list three types of random walk used in our spam detection experiments:

1. Following outlinks uniformly at random. Formally, define a random walk with
$$p(u,v) = \frac{w(u,v)}{d^+(u)}.$$

2. Following links uniformly at random regardless of directionality. Formally, define a random walk with
$$p(u,v) = \frac{w(u,v)+w(v,u)}{d^+(u)+d^-(u)}.$$

3. Following inlinks uniformly at random. Formally, define a random walk with

$$p(u,v) = \frac{w(v,u)}{d^-(u)}.$$

For other choices of random walks, we refer the readers to [31]. In our spam detection experiments, the third type of random walk achieves the best performance.

When we adopt the second type of random walk, its stationary distribution has the closed-form expression

$$\pi(v) = \frac{d(v)}{\sum_{u \in V} d(u)},$$

where $d(u) = d^+(u) + d^-(u)$. Substituting the expression into Equation (19), we have

$$\operatorname*{argmin}_{\phi \in \mathcal{H}(V)} \left\{ \sum_{u \sim v} a(u,v) \left(\frac{\phi(u)}{\sqrt{d(u)}} - \frac{\phi(v)}{\sqrt{d(v)}} \right)^2 + C \sum_{v \in V} (\phi(v) - y(v))^2 \right\}, \quad (20)$$

where $a(u,v) = w(u,v) + w(v,u)$. This is the classification approach proposed in [30]. There are several pieces of theoretic work on the statistical analysis of this approach [12, 1]. See also the approaches [17, 32, 3] which are closely related to that in [30].

Algorithm 1 Transductive Link Spam Detection

Given a web graph $G = (V, E)$, some web pages $S \subset V$ have been manually labeled as `normal` or `spam`. We assume the graph to be strongly connected. Otherwise, we decompose it into strongly connected components. The remaining unclassified web pages in V may be classified as follows:

1. Define a random walk which chooses an inlink uniformly at random to follow. Formally, this random walk has the transition probabilities

   $$p(u,v) = \frac{w(v,u)}{d^-(u)},$$

   for any u, v in V. Let π denote the vector which satisfies

   $$\sum_{u \in V} \pi(u) p(u,v) = \pi(v).$$

2. Denote by P the matrix with the elements $p(u,v)$, and Π the diagonal matrix with the diagonal elements $\pi(u)$. Form the matrix

   $$L = \Pi - \alpha \frac{\Pi P + P^T \Pi}{2},$$

   where α is a parameter in $]0,1[$.

3. Define a function y on V with $y(v) = 1$ or -1 if web page v is labeled as `normal` or `spam`, and 0 if v is unlabeled. Solve the linear system

   $$L\varphi = \Pi y,$$

   and classify each unlabeled web page v as $\operatorname{sign} \varphi(v)$.

For solving the optimization problem (18), differentiate the objective function with respect to φ and then obtain

$$\Delta_{\mathrm{am}} \varphi + C(\varphi - y) = 0,$$

where the first term on the left hand side is derived from Equation (11) via the differential rule on inner products. The above equation can be written as

$$(CI + \Delta_{\mathrm{am}})\varphi = Cy,$$

where I is the identity matrix. We can check that this linear system has the closed-form solution

$$\varphi = C(CI + \Delta_{\mathrm{am}})^{-1} y.$$

although it is more efficient to solve the linear system directly, rather than computing the inverse.

We summarize our final method in Algorithm 1 for the choice of random walk that inversely follows the links. In the algorithm, we use a parameter $\alpha \in]0,1[$ instead of $C \in]0,\infty[$. The relationship between α and C can be expressed as

$$\alpha = \frac{1}{1+C}.$$

Note that, in the last step of the algorithm, the classification is based on the sign of the function value on each vertex. This is equivalent to setting the classification threshold to 0. Since in practice it is much worse to classify a good web page as spam than to classify a spam web page as good, we suggest to set a threshold smaller than 0. In Section 5 we will show the details on evaluating the performance of a spam detection algorithm. In addition, for decomposing a directed graph into strongly connected components, we refer the readers to the depth-first search based approach [25].

6. EXPERIMENTS

We address the spam detection issue by using the dataset of `webspam-uk2006-1.2` [7]. This collection includes 77.9 million web pages over 11,452 hosts. They are labeled as `normal`, `borderline`, `spam`, and `cannot judge`. As in [6], for simplicity, we consider the spam detection issue at the host level. In other words, we consider if a host is spam or not. At the host level, 5.91% hosts are labeled as `spam`, and 43.45% hosts are labeled as `normal`. The remaining 50.69% hosts are `borderline` or `cannot judge`.

We can construct a directed graph over hosts as follows. Each host can be regarded a collection of web pages. Given two hosts, if there exists a hyperlink from some page on one host to some page on the other host, then we say that there is a directed edge between these two hosts. The edge is naturally weighted by the number of such edges. For a fair comparison among different approaches, we only consider the largest subgraph of the host graph in which each vertex is definitely labeled as `spam` or `normal`. In other words, we remove all vertices which are `borderline` or `cannot judge`. This subgraph contains 5,622 vertices. We then break it into strongly connected components. For a total, we obtain 1,332 components. The size of the largest one is 4,148, that is, the largest one contains 73.78% vertices. The second largest one contains only 21 vertices, and the third largest one 15 vertices. All of the remaining components contain less than 10 vertices. In fact, 96.61% of them contain a single vertex only. We choose the largest strongly connected component to compare different approaches.

Spam detection is a highly unbalanced classification issue. In the above chosen component, only 5.52% vertices

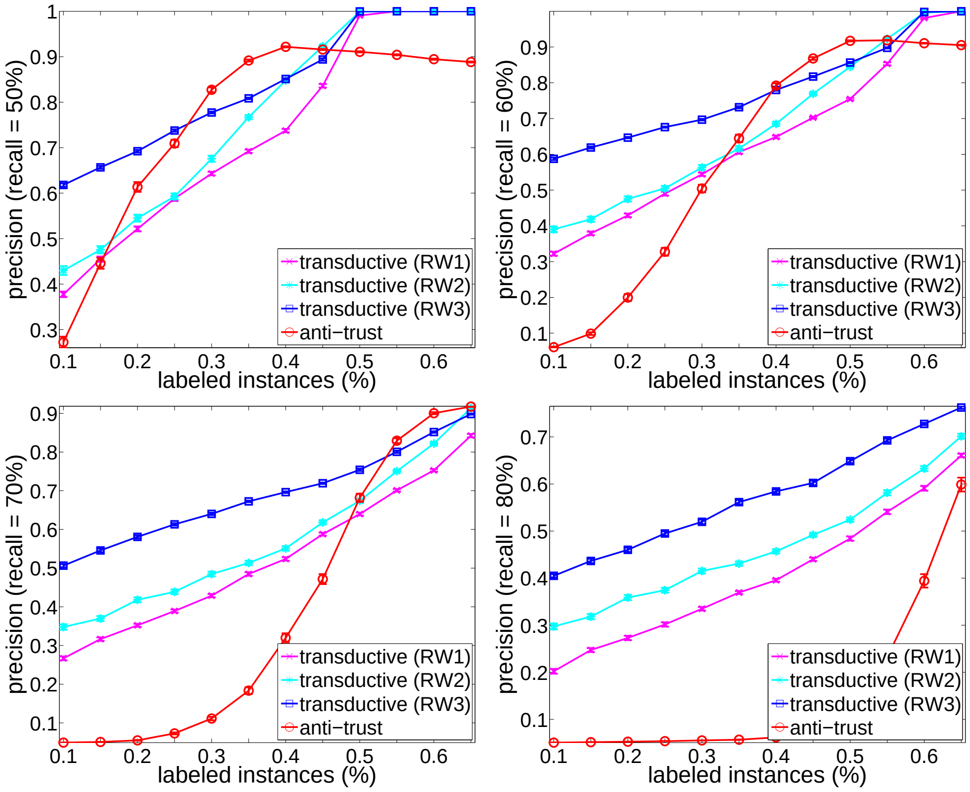

Figure 3: Precision vs. percentage of labeled instances

are labeled as `spam`. Hence, we measure algorithmic performances via precision/recall, rather than classification accuracy. Precision is the ratio of the number of retrieved and relevant documents to the number of documents retrieved, and recall is the proportion of the number of relevant documents that are retrieved to the total number of the relevant documents available. In addition, classifying a normal host into spam is much worse than classifying a spam host into normal. That means precision is more crucial than recall. Consequently, comparing precision with low recall is more significant than comparing precision with high recall.

We compare four different approaches. One is AntiTrust, and the others are the transductive classification approach with those three kinds of random walks presented in Section 5. The regularization parameter α is set to 0.15 as in [31]. The experimental results are summarized in Figures 3 and 4, where the transductive methods with different random walks are denoted as `transductive(RW1)`, `transductive(RW2)` and `transductive(RW3)`. Each result is averaged over 100 trials. In Figure 3, the proportion of labeled instances varies from 10% to 65% for both `spam` and `normal` examples. The precisions of all compared approaches are illustrated in four figures with a fixed recall respectively equal to $50\%, 60\%, 70\%$ and 80%. In Figure 4, the recall varies from 50% to 100%. The precisions of all compared approaches are illustrated in four figures with a fixed proportion of labeled instances respectively equal to $20\%, 30\%, 45\%$ and 50%.

The experimental results show that transductive classification approaches perform better than AntiTrust. This is because the transductive approaches can utilize both spam and normal instances while AntiTrust only utilizes spam instances. Unlike AntiTrust, TrustRank only utilizes normal instances. We also tested TrustRank in our experiments, and it is much worse than the above approaches, so we do not list the results from TrustRank. An obvious reason that AntiTrust and TrustRank do not work well is that some normal blog web sites are spammed. Specifically, a spam web site puts posts on the normal blog web sites, and then obtains links from the normal blog web sites to the spam web site. Consequently, AntiTrust will regard the blog web sites as spam, and TrustRank will regard the spam site as normal. If we can utilize information from both spam and normal examples, then these issues suffered by AntiTrust and TrustRank will be avoided.

The experimental results also show that, among all transductive approaches, the one based on the inverse random walk performs best. This can be explained by the observation that a web site which points to spam web sites is likely to be spam, while a web site pointed by spam web sites may or may not be spam. Consequently, it is meaningful to inversely follow an inlink of spam web sites for spam detection. Such a random walk is not good for detecting normal web sites. That is because a web site pointed by good web sites is likely to be good while a web site pointing to spam web sites may or may not be good. However, the weakness of the inverse random walk does not matter that much. It is pretty easy to obtain a large amount of normal web site examples, e.g. university and government web sites, and those normal examples will be fully exploited by the transductive methods. In contrast, it is very hard to judge a web site to be spam or not. Generally, we need trained human specialists

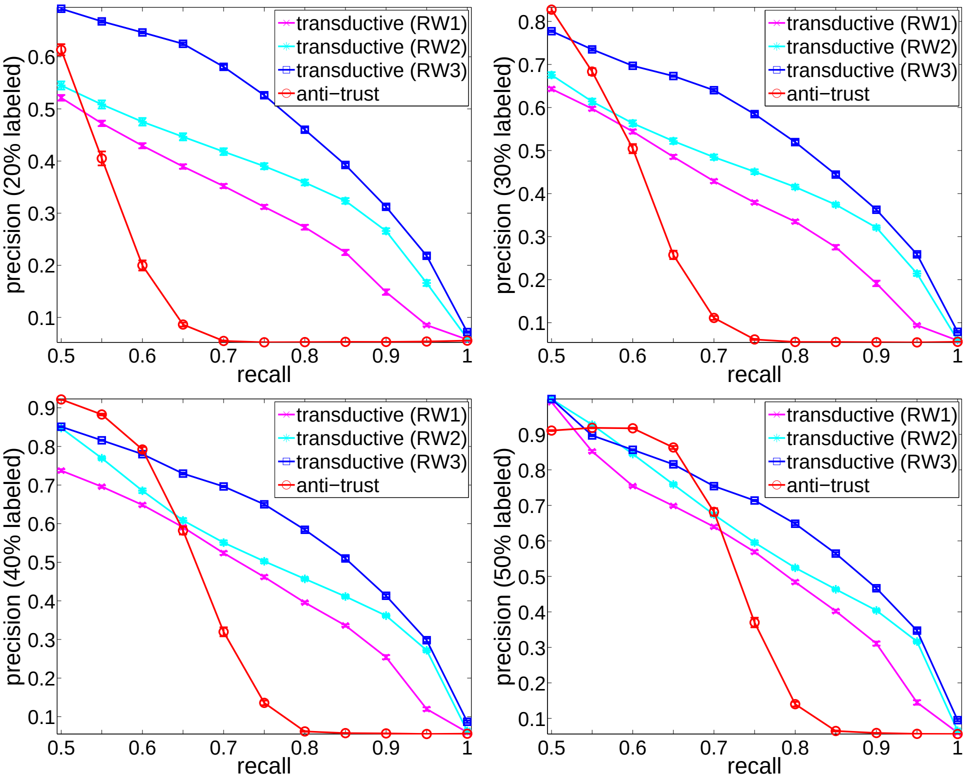

Figure 4: Precision vs. recall

to manually label web site as spam or normal.

7. DISCUSSION AND CONCLUSION

We proposed discrete analysis on directed graphs, and developed a discrete analog of classical regularization theory. A powerful classification method for directed graphs was derived from the discrete regularization, and it was validated in real-world spam detection tasks. There are a number of interesting further directions suggested by this work. We will restrict ourselves to three such directions.

First, so far, the directed graphs that we considered are required to be strongly connected. If a graph is not strongly connected, then we need to break it into strongly connected components, and consequently carry out the classification task on each component. Hence, our approach needs both positive and negative labeled instances in each component. One would suggest to overcome this issue by adopting the random walk used in PageRank instead of the natural rank walk used here. It is well-known that the random walk used in PageRank converges to a unique stationary distribution. However, this random walk does not work well in our experiments. Another solution to remedying this issue is to explore some heuristics which can effectively propagate the labeling information for the largest strongly connected components to others. As investigated in [5, 20], in the web graph, there is a single large strongly connected component, and all other components are significantly smaller in size. They also show that the distribution of the sizes of strongly connected components obeys a power law. From this observation, we mainly need to detect spam in the largest strongly connected component. Once this component is fully labeled, then the remaining much smaller components can be labeled via some label propagation from the largest one. Such a heuristic may work well. However, we believe that there should exist a principled way to address this issue.

Second, it's worth exploring how to choose some web sites to manually label, such that they are more helpful in our spam detection than those web sites which are randomly chosen for labeling. Both AntiTrust and TrustRank have addressed similar issues of how to choose those normal or spam web sites to obtain better performance. How to choose the most helpful web sites to label is an important issue in practice because finding spam web sites requires expensive human labor. As we have mentioned, each precision point in Figures 3 and 4 is averaged over 100 trials. We notice that the precisions in some trials are much higher then other trials. We can formally describe the issue of selective labeling as follows. Given a directed graph $G = (V, E, w)$ and a number $k < |V|$, we can take the labels of any k vertices and our task is to predict the labels of the remaining vertices as accurately as possible. We can call it active learning for directed graphs. So far we do not see any literature on investigating this problem. However, there is much literature on active learning for inductive algorithms (e.g., see [10, 24]). Those might by helpful for developing active learning algorithms on directed graphs.

Third, our discrete regularization is built on the discrete gradient operator which is first-order differential. It is obvious that the first-order differential is not powerful enough in illustrating the smoothness of a function. In classical regularization theory, arbitrary order differentials have also been considered. In kernel methods, if we use the Gaussian kernel, then the differential at any order will be involved [23]. For developing a complete discrete analog of classical regularization and kernel theory for graphs, we have to first develop the discrete analog of high-order differentials, like Hessian, connection and curvatures. Regarding this way of thinking about machine learning issues on graphs or discrete sets, our research is still at the very beginning.

8. ACKNOWLEDGMENTS

We would like to thank Kumar Chellapilla of Microsoft Live Labs for his help on this work.

9. REFERENCES

[1] A. Ando and T. Zhang. Learning on graph with Laplacian regularization. In *Advances in Neural Information Processing Systems 19*, Cambridge, MA, 2007. MIT Press.

[2] R. Baeza-Yates, P. Boldi, and C. Castillo. Generalizing PageRank: Damping functions for link-based ranking algorithms. In *Proc. 29th Annual International ACM SIGIR Conference on Research & Development on Information Retrieval*, pages 308–315, 2006.

[3] M. Belkin, I. Matveeva, and P. Niyogi. Regression and regularization on large graphs. In *Proc. 17th Annual Conference on Learning Theory*, 2004.

[4] S. Brin and L. Page. The anatomy of a large scale hypertextual web search engine. In *Proc. 7th International World Wide Web Conference*, 1998.

[5] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the Web. In *Proc. 9th International World Wide Web Conference*, 2000.

[6] C. Carlos, D. Debora, G. Aristides, M. Vanessa, and S. Fabrizio. Know your neighbors: Web spam detection using the web topology. Technical report, 2006.

[7] C. Castillo, D. Donato, L. Becchetti, P. Boldi, M. Santini, and S. Vigna. A reference collection for web spam. *SIGIR Forum*, 40(2), 2006.

[8] F. Chung. Laplacian and the Cheeger inequality for directed graphs. *Annals of Combinatorics*, 9:1–19, 2005.

[9] F. Chung, A. Grigoryan, and S.-T. Yau. Higher eigenvalues and isoperimetric inequalities on Riemannian manifolds and graphs. *Communications on Analysis and Geometry*, 8:969–1026, 2000.

[10] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145, 1996.

[11] S. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the Society for Information Science*, 41(6):391–407, 1990.

[12] R. El-Yaniv and D. Pechyony. Stable transductive learning. In *Proc. 19th Annual Conference on Computational Learning Theory*, pages 35–49, 2006.

[13] Z. Gyongyi, H. Garcia-Molina, and J. Pedersen. Combating web spam with trustrank. In *Proc. 30th International Conference on Very Large Data Bases*, pages 576–587, 2004.

[14] T. H. Haveliwala. Topic-sensitive pagerank. In *Proc. 11th International World Wide Web Conference*, pages 517–526, 2002.

[15] M. Hein, J. Audibert, and U. von Luxburg. From graphs to manifolds - weak and strong pointwise consistency of graph Laplacians. In *Proc. 18th Annual Conference on Learning Theory*, pages 470–485, 2005.

[16] M. Henzinger. Hyperlink analysis for the web. *IEEE Internet Computing*, 5(1):45–50, 2001.

[17] T. Joachims. Transductive learning via spectral graph partitioning. In *Proc. 20th IInternational Conference on Machine Learning*, 2003.

[18] J. Jost. *Riemannian Geometry and Geometric Analysis*. Springer-Verlag, Berlin-Heidelberg, third edition, 2002.

[19] J. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

[20] J. Kleinberg, R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. The web as a graph: Measurements, models, and methods. In *Proc. 5th International Conference on Computing and Combinatorics*, 1999.

[21] A. Ntoulas, M. Najork, M. Manasse, and D. Fetterly. Detecting spam web pages through content analysis. In *Proc. 15th International World Wide Web Conference*, 2006.

[22] R. Raj and V. Krishnan. Web spam detection with anti-trust rank. In *Proc. 2nd International Workshop on Adversarial Information Retrieval on the Web*, pages 37–40, 2006.

[23] B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[24] T. Simon and D. Koller. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66, 2001.

[25] R. Tarjan. Depth first search and linear graph algorithms. *SIAM Journal on Computing*, 1:146–160, 1972.

[26] A. Tikhonov and V. Arsenin. *Solutions of Ill-posed Problems*. W. H. Winston, Washington, DC, 1977.

[27] V. Vapnik. *Statistical Learning Theory*. Wiley, NY, 1998.

[28] G. Wahba. *Spline Models for Observational Data*. Number 59 in CBMS-NSF Regional Conference Series in Applied Mathematics. SIAM, Philadelphia, 1990.

[29] B. Wu and B. D. Davison. Identifying link farm spam pages. In *WWW (Special interest tracks and posters)*, pages 820–829, 2005.

[30] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

[31] D. Zhou, J. Huang, and B. Schölkopf. Learning from labeled and unlabeled data on a directed graph. In *Proc. 22th International Conference on Machine Learning*, 2005.

[32] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *Proc. 20th International Conference on Machine Learning*, 2003.