Inf Retrieval J (2017) 20:547–574
DOI 10.1007/s10791-017-9307-9

A unified score propagation model for web spam demotion algorithm

Xu Zhuang[1] · Yan Zhu[1] · Chin-Chen Chang[2,3] · Qiang Peng[1] · Faisal Khurshid[1]

Received: 26 April 2016 / Accepted: 29 April 2017 / Published online: 10 May 2017

Abstract Web spam pages exploit the biases of search engine algorithms to get higher than their deserved rankings in search results by using several types of spamming techniques. Many web spam demotion algorithms have been developed to combat spam via the use of the web link structure, from which the goodness or badness score of each web page is evaluated. Those scores are then used to identify spam pages or punish their rankings in search engine results. However, most of the published spam demotion algorithms differ from their base models by only very limited improvements and still suffer from some common score manipulation methods. The lack of a general framework for this field makes the task of designing high-performance spam demotion algorithms very inefficient. In this paper, we propose a unified score propagation model for web spam demotion algorithms by abstracting the score propagation process of relevant models with a forward score propagation function and a backward score propagation function, each of which can further be expressed as three sub-functions: a splitting function, an accepting function and a combination function. On the basis of the proposed model, we develop two new web spam demotion algorithms named Supervised Forward and Backward score Ranking (*SFBR*) and Unsupervised Forward and Backward score Ranking (*UFBR*). Our experiments, conducted on three large-scale public datasets, show that (1) *SFBR* is very robust and apparently outperforms other algorithms and (2) *UFBR* can obtain results comparable to some well-known supervised algorithms in the spam demotion task even if the *UFBR* is unsupervised.

✉ Xu Zhuang
zhuangxusc@gmail.com

[1] School of Information Science and Technology, Southwest Jiaotong University, Chengdu 610031, Sichuan, China

[2] Department of Information Engineering and Computer Science, Feng Chia University, Taichung 40724, Taiwan

[3] Department of Computer Science and Information Engineering, Asia University, Taichung 41354, Taiwan

Keywords Web spam demotion · Web ranking algorithms · Spam detection · Web scoring system

1 Introduction

Currently, people rely increasingly more on search engines (Hochstotter and Koch 2009) to retrieve useful information from the growing mountain of data on the web. A user sends a query containing one or more keywords to a search engine, and the search engine responds with a long search result list, with pages sorted in descending order by their relevance and importance. Previous studies have shown that, for 85% of queries, only the first search engine result page is requested, and on average, only the first three to five links are clicked (Silverstein et al. 1999). Therefore, a higher ranking in search engine results corresponds to greater economic benefits due to the referral traffic from search engines. To obtain a higher ranking in search results, several types of spamming techniques (Becchetti et al. 2008; Chellapilla and Chickering 2006; Cormack et al. 2011; Ntoulas et al. 2006; Radlinski 2007) are used by spammers to deceive search engine algorithms. A web page is termed *web spam* if it uses any spamming technique to trigger an unjustified ranking in search engine results (Al-Kabi et al. 2012; Gyongyi and Garcia-Molina 2005). Web spam was recognized as a key challenge for search engine companies (Henzinger et al. 2002) soon after the concept was first invented in 1996 (Convey 1996). First, web spam causes a sharp drop in search result quality and hence reduces the efficiency and reputation of search engine companies. Second, spam sites squeeze out their competitors with undue competition to deprive normal websites of revenue that they might earn in the absence of web spam (Spirin and Han 2012). Third, web spam is an important channel for the spread of malware and adult content and thus leads customers to suffer from malicious attacks and even economic losses. Fourth, the low quality of search results due to web spam leads to poor user experiences, causing customers to spend more time and effort to find their desired web sites, which may in turn result in customer churn for search engine companies since the cost of switching from one search provider to another is nearly zero (Spirin and Han 2012). Lastly, web spam causes great economic losses to search engine companies because of the extra resource costs of storing, indexing and analysing spam pages.

Many web spam demotion algorithms (*WSDA*s) have been proposed to combat spam by demoting their rankings in search engine results. Compared to conventional machine learning-based methods (Becchetti et al. 2006, 2008; Chandra et al. 2015; Ntoulas et al. 2006) that aim to detect spam, *WSDA*s have some obvious advantages. *WSDA*s use only the link structure of web pages, and constructing a link-based web graph is much easier than obtaining all of the HTML of web pages for content analysis in terms of both time and space consumption. Furthermore, a *WSDA* can achieve results comparable to those of a supervised machine learning algorithm with much fewer labelled samples. Thus, *WSDA*s can be applied to practical applications in which very few pages are labelled in general.

Most of the published *WSDA*s are based on the PageRank model (Page et al. 1999), in which the "global importance" of each web page is estimated on the basis of the web graph. With a similar idea, TrustRank (Gyöngyi et al. 2004) propagates trust scores through hyper-links from a set of trusted seeds (i.e., normal pages) to evaluate the goodness of web pages, as links between pages are considered an implication of trust. TrustRank relies on an empirical observation called approximate isolation of the good set: good pages

seldom point to bad pages (Gyöngyi et al. 2004). Although TrustRank can largely demote web spam with little assistance from humans, its spam-demoting effectiveness decreases with the rise of the number of good-to-bad links. Anti-Trust Rank (Krishnan and Raj 2006) follows another intuition of the approximate isolation principle: it is very unlikely for spam pages to be pointed to by good pages (Krishnan and Raj 2006). This means that a page is highly likely to be spam if it points to many other spam pages. Thus, distrust is convened in the inverse link from spam pages. On the basis of a bad seed set (i.e., spam pages), Anti-TrustRank propagates distrust through inverse links of the web graph to evaluate the badness of web pages. To take advantage of TrustRank and Anti-Trust Rank, Wu et al. (2006) proposed LCRank to simply combine these two algorithms by linear fusion, and they obtained slightly better results. Subsequently, several schemes conducting differential score propagation were proposed, including TDR (Zhang et al. 2011) and GBR (Liu et al. 2013). Differential score propagation methods focus on designing a discriminatory score propagation process for source (score propagated from) or target (score propagated to) web pages. These types of score propagation methods obtained state-of-the-art performance by propagating trust and anti-trust simultaneously and making use of the good and bad scores of pages in the propagation process.

In this paper, we propose a unified score propagation model (*USPM*) for *WSDA*s. We show that all *WSDA*s mentioned in the paper are specific implementations of the model and that they differ from their base model via only very limited improvements. Utilizing the proposed *USPM*, it is very easy to analyse differences and drawbacks of previous models and hence guide researchers to design more complex and effective schemes. In summary, our contributions in this paper are as follows:

(1) We formulate web spam demotion algorithm and develop the *USPM* by abstracting the unified score propagation behaviours of different models with a forward score propagation function (*FSPF*) and a backward score propagation function (*BSPF*), each of which can further be expressed as three sub-functions: a splitting function, an accepting function and a combination function. Different *WSDA*s can be thought of as different combinations of these sub-functions.
(2) We summarize and analyse various published candidates, as well as introduce new strategies for sub-functions of *FSPF* and *BSPF*.
(3) We adopt the normalized technique to overcome the loss of effectiveness of the distribution vector of inconsistent[1] *WSDA*s.
(4) We formulate metrics to evaluate the abilities of *WSDA*s to demote and detect spam.
(5) We propose the first unsupervised *WSDA* by taking advantages of functions introduced in the paper.
(6) We experimentally demonstrate the superiority of the proposed *SFBR* algorithm and show promising results obtained by *UFBR* in some cases.

The remainder of this paper is organized as follows. Section 2 formulates the unified score propagation model, *USPM*. In Sect. 3, different *WSDA* strategies are presented. Our new algorithms are proposed in Sect. 4. Section 5 concerns the experiments and discussions. Finally, conclusions are drawn in Sect. 6.

[1] A *WSDA* is termed inconsistent if in-system scores will be lost or out-system scores will be brought in during the score propagation process.

2 Unified score propagation model

The web is modelled as a directed web graph $\mathcal{G}$ consisting of a set of nodes (web pages or sites) $\mathcal{N}$ and a set of directed edges $\mathcal{E}$. An edge $(p, q) \in \mathcal{E}$ if there exist hyper-links from page p to q. Self-links are removed, and multiple links between pages are treated as a single link to simplify the model. The sets of incoming nodes (pages pointing to p) and outgoing nodes (pages pointed to by p) of a page p are denoted as $In(p)$ and $Out(p)$, respectively. Page p is a *parent* of its outgoing nodes and a *child* of its incoming nodes. The notation $|S|$ denotes the number of elements in set S. For example, $|\mathcal{N}|$ is the total number of nodes in the web graph $\mathcal{G}$.

As indicated in the approximate isolation principle (Marchiori 1997), both trust and distrust are conveyed in links: trust is propagated along the link direction, while distrust is propagated along the inverse link direction. Thus, trust and distrust can be used to estimate the goodness and badness of a web page. More generally, each web page in the web graph is assigned two scores in the proposed Unified Score Propagation Model (*USPM*), the forward score (*FS*) and the backward score (*BS*), which are propagated through forward links and backward links, respectively. The forward score represents the likelihood of a web page being good, and the backward score represents the likelihood of a web page being bad. Note, however, that *FS* or *BS* cannot provide an absolute probability of a web page being good or bad. We hope that those scores could at least help us rank web pages, which is called the ordered property of *FS* and *BS*.

Formally, we define the *state* of a web page p as a pair of non-negative variables, its forward score $FS(p)$ and backward score $BS(p)$, denoted as $ST(p) = (FS(p), BS(p))$. *A web spam demotion algorithm computes the state of every node in the web graph.* Having the states of all of the nodes, we can order pages by their *FS* to demote spam or by their *BS* to detect spam (see Sect. 5 for details).

Next, we introduce a simplified *WSDA* framework on web graph $\mathcal{G}$ by walking through its score computation steps.

(1) *Score initialization* The state of every node in the web graph is initialized with assigned *FS* and *BS*. The simplest method for this is to assign uniform values of *FS*s and *BS*s to all nodes, e.g., $1/|\mathcal{N}|$.

(2) *Score propagation* In a single score propagation iteration, every node propagates *FS*s to its children and *BS*s to its parents. A *splitting function* is used to calculate the *FS* or *BS* that should be propagated from a source node. The *FS* propagated from node p to its children is calculated by the *forward score splitting function* $f^1_{split}(p)$, and the *BS* propagated from node p to its parents is calculated by the *backward score splitting function* $f^2_{split}(p)$. Once node p receives an *FS* (denoted as *fs*) from its parent (say k), the *forward score accepting function* $f^1_{accept}(p, fs)$ is used to determine the accepted score from *fs*, where $fs = f^1_{split}(k)$. Similarly, the *backward score accepting function* $f^2_{accept}(p, bs)$ is used to determine the accepted score from $bs = f^2_{split}(q)$, where q is p's child. Taking advantage of accepting functions, node p can obtain a set of accepted *FS*s (denoted as S_{fs}) from all of its parents and a set of *BS*s (denoted as S_{bs}) from all of its children. Finally, node p uses the *forward score combination function* to fuse *FS*s in S_{fs} and assigns its own *FS* with the result $f^1_{com}(S_{fs})$. Similarly, node p uses the *backward score combination function* to fuse *BS*s in S_{bs} and assigns its own *BS* with the result $f^2_{com}(S_{bs})$.

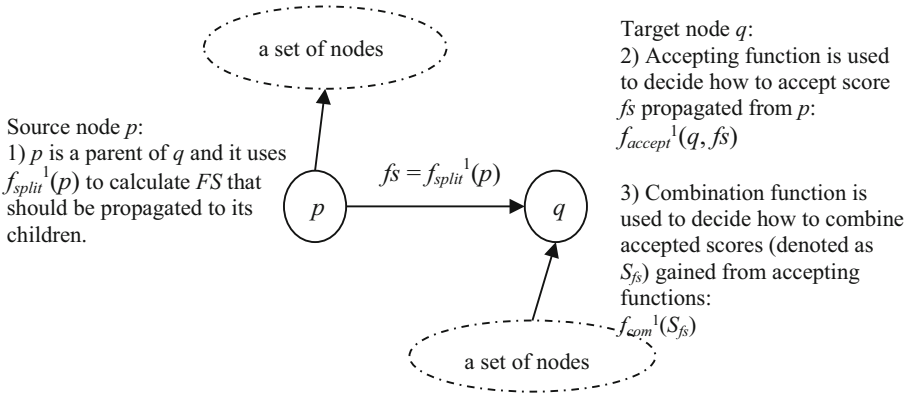

Fig. 1 Score propagation process of the forward score

As an example, the score propagation process of the forward score is described in Fig. 1. It shows that the functionalities and responsibilities of the source page (score propagated from) and target page (score propagated to) are separated in our model. The source page (e.g., p in Fig. 1) concentrates on how to split its FS to every child (e.g., q in Fig. 1), while the task of the target page is twofold: how to accept scores propagated from parents and how to combine them. In the model, every function has its own responsibility and is independent from others. Such a property allows researchers to focus on the parts in the model that they are most interested in, and hence, they can define their own substitutes for those functions to develop custom *WSDA*s.

We define the *forward score propagation function* (*FSPF*) to calculate the *forward score* (*FS*) of a web page p:

$$FS(p) = f_{com}^{1}(S_{fs}), \tag{1}$$

where

$$S_{fs} = \left\{ s | s = f_{accept}^{1}(p, f_{split}^{1}(q)), q \in In(p) \right\}. \tag{2}$$

In a similar manner, *BSPF* is defined:

$$BS(p) = f_{com}^{2}(S_{bs}), \tag{3}$$

where

$$S_{bs} = \left\{ s | s = f_{accept}^{2}(p, f_{split}^{2}(q)), q \in Out(p) \right\}. \tag{4}$$

Formulas (1) and (3) present simplified definitions of *FSPF* and *BSPF* since both the *FS* and *BS* of a node are fully determined by its incoming and outgoing nodes. In the random surfer model on the web, forward scores can be thought of as corresponding to the standing probability distribution of a random walker who is visiting web pages by simply clicking on successive links at random (Page et al. 1999; Diligenti et al. 2004). However, if a real web surfer falls into a loop of web pages, it is unreasonable to limit the behaviour of the surfer in such a way since the surfer could jump to some other web pages even if there is no direct link between the source page and the target page. An additional distribution vector is used to model this behaviour: the surfer can jump to a randomly selected web page based on the distribution vector. Therefore, the

total *FS* and *BS* of a web page can be gained from two independent mechanisms, namely, the score propagation process and the fixed distribution vector.

Having the distribution vector *dv* such that $dv(p)$ denotes the corresponding value of page p in *dv*, *FSPF* and *BSPF* can be rewritten as:

$$FS(p) = (1 - \alpha_1) \cdot f^1_{com}(S_{fs}) + \alpha_1 \cdot dv_1(p), \tag{5}$$

$$BS(p) = (1 - \alpha_2) \cdot f^2_{com}(S_{bs}) + \alpha_2 \cdot dv_2(p), \tag{6}$$

where α_1 and α_2 represent the jump probabilities.

The distribution vector is typically used to represent prior knowledge about labels (spam or normal) of web pages in most supervised *WSDA*s. For example, we can set $dv_1(p)$ to a very high value in the distribution vector if page p is identified as normal by human experts, and we hope that the priori information can help the score propagation process in such a way that p should always be assigned a high *FS* and pages directly pointed to by p can also obtain a high *FS* from p since p is trustworthy.

Consider a *WSDA* for which only the *FS* is propagated with the initialization: (1) the total *FS* equals 1, and (2) the L_1 norm of the distribution vector is 1. It is shown in Eq. (5) that the *FS* of a node consists of two parts, namely, one portion (determined by $1 - \alpha_1$) from the score propagation and the other (determined by α_1) from the distribution vector. The total *FS* will be consistent (1 in the example) if there are no out-system scores added in or in-system scores lost during the score propagation process. A function is *score consistent* if it does not bring in out-system scores or lose in-system scores. A *WSDA* is consistent if its total *FS* and *BS* are constants. Obviously, if all sub-functions are consistent, the *WSDA* is consistent. An inconsistent total score of a *WSDA* causes *effectiveness loss of the distribution vector*. For example, suppose that the total *FS* of a *WSDA* is amplified two times after each iteration. That means that the *FS* of a page contributed from the score propagation process will become increasingly larger, which in turn decreases the effectiveness of the distribution vector. Finally, the *FS* of a page is dominated by the score propagation process, and the effectiveness of the distribution vector is totally lost. To overcome this problem, we introduce a score normalization mechanism that is carried out after each score propagation iteration of a *WSDA*.

In summary, the complete *WSDA* framework is described as follows.

(1) *Score initialization* The state of every node in the web graph is initialized with assigned *FS* and *BS*.
(2) *Distribution vector initialization* Elements of the distribution vector are initialized in a predefined way.
(3) *Score propagation* In a single score propagation iteration, every node propagates *FS*s to its children and *BS*s to its parents.
(4) *Score normalization* is carried out to regularize scores in the system after each score propagation iteration, i.e., to maintain the total *FS* and *BS* as constant. Repeat steps (3) and (4) until algorithm convergence.

3 WSDA strategies

In this section, we first discuss key features of different *WSDA* sub-functions, including published strategies and some new ones introduced here. For convenience, only sub-functions of the forward score propagation function are discussed since all of these functions can also be applied to the backward score propagation function with minor changes. Then, the methods for seed selection and jump probability setting are presented.

3.1 Splitting function for FSPF

A splitting function $f^1_{split}(p)$ is used to determine the forward score propagated from parent p to its children.

3.1.1 Uniform splitting (Page et al. 1999)

The parent p uniformly divides its FS to its children:

$$f^1_{split}(p) = \frac{FS(p)}{|Out(p)|}. \tag{7}$$

The FS propagated from a page p is inversely proportional to the number of its outgoing nodes. This function models the lack of ability of a page to maintain its link qualities if it has a large number of outgoing links. The more outgoing links a page has, the higher the probability of the page pointing to spam. Hence, the propagated FS of a page decreases as the number of its outgoing links increases. Uniform splitting was put forward in PageRank (Page et al. 1999) for the first time and was widely adopted in many other algorithms (Gyöngyi et al. 2004; Krishnan and Raj 2006; Page et al. 1999; Wu et al. 2006). However, Wu et al. (2006) noted the issue caused by uniform splitting: why should equally trusted pages (e.g., pages with the same FSs) propagate different FSs to their children just because they have different numbers of outgoing links? This is alarming in the sense that less trusted pages may propagate more scores than trusted pages due to their very few outgoing links. For example, page p is pointed to by pages A and B, which have different FSs of 0.1 and 0.01, respectively. However, A might propagate a lower FS than B to p if A has more than ten times the number of outgoing links as B, even though A is more likely to be good.

Uniform splitting is particularly vulnerable to link farm spamming that a set of spam pages are generated to boost the FS of a target page by pointing to it. Pages in a link farm can propagate a considerable total FS to the target page due to their very few outgoing links. Thus, uniform splitting is not a good choice to demote spam for its bad anti-spamming ability.

3.1.2 Logarithm splitting (Wu et al. 2006)

Logarithm splitting is a variant of uniform splitting, and it aims to decrease the effect of the number of out-going links:

$$f^1_{split}(p) = \frac{FS(p)}{\log(1 + |Out(p)|)}. \tag{8}$$

3.1.3 Attenuation splitting (Wu et al. 2006)

$$f^1_{split}(p) = d \times FS(p), \quad 0 < d < 1. \tag{9}$$

Attenuation splitting is designed to address the problem caused by uniform splitting (see Sect. 3.1.1) such that the *FS* propagated from the source node is independent of its number of out-links. The decay factor d is used to model the fact that the further away a page is from the good seed set, the less certain it is that the page is good. Obviously, attenuation splitting is inherently biased to pages near the seeds since they can obtain higher scores than pages that are far from the seeds.

3.1.4 Constant splitting

The *FS* propagated from page p to its child q is equal to p's *FS* without attenuation:

$$f^1_{split}(p) = FS(p). \tag{10}$$

Constant splitting is simple but can solve problems caused by uniform splitting and attenuation splitting. With constant splitting, (1) pages with higher *FS*s can propagate higher scores to their children; (2) the high *FS*s of seeds can propagate to remote nodes, and hence, the model's sensitivity to the seed set is greatly decreased. Although spam pages can benefit from good-to-bad links in such cases, referring to the approximate isolation principle that good pages seldom point to bad pages, very few spam pages can obtain *FS*s from good pages. However, constant splitting suffers from the problem of score bombing such that the total *FS*s of all nodes largely increase after every iteration, which causes difficulty in maintaining stable running of the algorithm.

3.1.5 Linear fusion

$$f^1_{split}(p) = \begin{cases} \beta \cdot FS(p) - (1-\beta) \cdot BS(p), \ \beta \cdot FS(p) > (1-\beta) \cdot BS(p) \\ 0, \ \beta \cdot FS(p) \le (1-\beta) \cdot BS(p) \end{cases}, \quad 0 < \beta < 1. \tag{11}$$

The *FS* propagated from page p to its child is penalized by its gained *BS*. Formula (10) is quite similar to the fusion function used in LCRank (Wu et al. 2006). However, there are some apparent differences between these two schemes. *FS* and *BS* are calculated separately in LCRank, and they are combined after algorithm convergence. LCRank can be seen as a fusion scheme of the ranking results of two algorithms (i.e., TrustRank and Anti-TrustRank), whereas *FS* and *BS* are actually combined in the splitting process of linear fusion. The parameter β is used to adjust the weights of *FS* and *BS* in the linear fusion.

3.1.6 Proportional splitting

$$f^1_{split}(p) = \begin{cases} Base(p) \times \dfrac{\beta \cdot FS(p)}{\beta \cdot FS(p) + (1-\beta) \cdot BS(p)}, FS(p) \neq 0 \\ Base(p), \ FS(p) = 0 \end{cases}, \quad 0 \le \beta \le 1, \tag{12}$$

where the base score function $Base(p)$ is based on the current node p and its value is called the base score. A web page is described by two different aspects in our model: the good

side and bad side, which are evaluated by *FS* and *BS*, respectively. The part of Formula (12) $\beta \cdot FS(p)/(\beta \cdot FS(p) + (1-\beta) \cdot BS(p))$ is used to calculate the likelihood of a web page p being good based on its current state. That likelihood is then used as a decay factor (called the proportional decay factor) for the base score. The parameter β is introduced to adjust the weights of *FS* and *BS* in Formula (12).

TDR and GBR experimentally showed the effectiveness of the proportional decay factor with the base score calculated by the uniform splitting function in Eq. (7). However, TDR and GBR simply inherit the computation method for the base score from TrustRank, and hence, we doubt its effectivity in demoting spam because the uniform splitting has some obvious weaknesses. We investigate other base score functions for the proportional splitting function in Sect. 4.

3.2 Accepting function for FSPF

An accepting function, $f^1_{accept}(p, fs)$, is used to determine the score that should be accepted by target node p on the basis of the score fs propagated from p's parent q, where fs is calculated through the splitting function $f^1_{split}(q)$.

3.2.1 Constant accepting (Page et al. 1999)

$$f^1_{accept}(p,fs) = fs. \tag{13}$$

The propagated score fs from the parent is accepted by the child without any change. Constant accepting is widely used in many algorithms (Gyöngyi et al. 2004; Krishnan and Raj 2006; Liu et al. 2013; Page et al. 1999) for its simplicity. However, the effect of the constant accepting function has not been well studied.

3.2.2 Proportional accepting

$$f^1_{accept}(p,fs) = \begin{cases} fs \times \dfrac{\beta \cdot FS(p)}{\beta \cdot FS(p) + (1-\beta) \cdot BS(p)}, & FS(p) \neq 0 \\ fs, FS(p) = 0 \end{cases}, \tag{14}$$

where $0 \leq \beta \leq 1$. The portion of *FS* in the child node is computed and then used to damp the final accepted score from the propagated fs. The parameter β is introduced to adjust the weights of *FS* and *BS* in the function.

3.2.3 Uniform accepting

$$f^1_{accept}(p,fs) = \frac{fs}{|In(p)|}. \tag{15}$$

The accepted score is decayed using the number of incoming links of the child node. As a consequence, the page accepts small *FS*s if it has many incoming links. This method may be useful to combat link-farm spamming because the qualities of parents are emphasized instead of the quantity.

3.2.4 Logarithm accepting

$$f^1_{accept}(p,fs) = \frac{fs}{\log(1 + |In(p)|)}. \tag{16}$$

In the web environment, a page cannot manage incoming links itself. Uniform accepting can address the problem caused by link farm spamming but greatly decreases the *FS* of a good page that has many incoming links, especially for a very popular web site, such as Google or Yahoo!.

Logarithm accepting has similar effectiveness as uniform accepting but decreases the effect of the number of incoming links of the target node.

3.3 Combination function for FSPF

A child node can obtain a set of accepted *FS*s (denoted as a set *S*) from its parent nodes. A combination function for *FSPF* is used to determine how to combine scores in *S*.

3.3.1 Simple summation combination (Page et al. 1999)

$$f^1_{combination}(S) = \sum_{e \in S} e. \tag{17}$$

Scores in *S* are simply accumulated. All algorithms (Gyöngyi et al. 2004; Krishnan and Raj 2006; Liu et al. 2013; Page et al. 1999; Wu et al. 2006; Zhang et al. 2011) adopt this method to combine accepted forward scores. It is known that PageRank and its variants are inherently biased to two types of web pages: (1) pages with many parent pages and (2) pages with parent pages that have high *FS*s. By defining the combination function in our model, it is obvious that such biasness is largely due to the simple summation combination function. A page receives the total summation of forward scores propagated from its parents so that the page obtains a high forward score if it has many parents, even though all of them have small *FS*s. In such a case, a large quantity of parent pages compensates for their low qualities. Web spam is so rampant because spammers can utilize this defect to "transform" quantity to quality. Moreover, it is quite easy for spammers to do so by creating a large number of incoming links for their own pages, i.e., link farm spamming.

3.3.2 Maximum share combination (Wu et al. 2006)

$$f^1_{combination}(S) = Max(S), \tag{18}$$

where $Max(S)$ denotes the maximum element in set *S*.

The maximum value in *S* is selected as the final *FS* of the node. This method can eliminate the effect of the number of incoming links. A page obtains a high *FS* if and only if it has at least one parent page with a high *FS*.

3.3.3 Maximum parent combination (Wu et al. 2006)

$$f^1_{combination}(S) = Min\left(\left\{\sum_{e\in S} e, a\right\}\right), \tag{19}$$

where $a = Max(\{FS(q)|q \in IN(p)\})$, p is the current node and $Min(S)$ denotes the minimum element in set S.

The simple summation is first performed, and then, its value is compared with the FS of p's most-trusted parent. The smaller of these two values is assigned to p. In other words, the node p can obtain a score as high as possible from the set S with the constraint that it never exceeds the FS of its most-trusted parent.

3.3.4 Top-n score combination

$$f^1_{combination}(S) = \sum_{i=1}^{n} S_{sort}(i), \tag{20}$$

where S_{sort} denotes the result set by sorting S in descending order and $S_{sort}(i)$ represents the i-th largest element in S_{sort}.

The top n largest elements in S are accumulated. The trade-off between quantity and quality can be adjusted by using different n. A larger n implies that most of the elements in S will be accumulated, where quantity has higher priority than quality. A smaller n means that only those most-trusted scores in S will be considered to determine the FS of the target page.

All of the above functions used for $FSPF$ can also be applied to $BSPF$ with minor changes. Thus, sub-functions for $BSPF$ are not discussed again because of their similar meanings to functions for $FSPF$.

3.4 Seed selection methods

As discussed in Sect. 2, the typical usage of the distribution vector is to enable the use of a priori labelling information of web pages for a more effective score propagation process. In general, we expect a small seed set of pages that can be labelled by human experts within a feasible amount of time and hope that the seed set can improve the performance of the algorithm. Several heuristics have been proposed to select the seed set. One popular approach for good seeds is to select pages that point to many other pages. Thus, the FSs of seed pages can be widely spread to others. To achieve this goal, the Inverse-PageRank is introduced to select good seeds in [4] (Gyöngyi et al. 2004). Inverse-PageRank utilizes only $BSPF$, and it evaluates the "importance" of a web page by investigating its outgoing links. Pages will be assigned high Inverse-PageRank scores if they point to many other pages or if pages pointed to by them have high Inverse-PageRank scores. The method is propitious to fast propagation of FSs in the graph since seed pages are likely to point to others.

Another common method is to select pages with high PageRank scores because those pages will appear high in query result lists, and hence, it is more important for search engine to ascertain the goodness of those pages [4] (Gyöngyi et al. 2004). In this paper, we use the PageRank algorithm to select good seeds. Via a similar idea, the bad seed set is selected by Inverse-PageRank since it is important for a search engine that aims to find bad pages to ascertain the badness of pages with high positions in result lists.

3.5 Jump probability

The jump probability is also known as the decay factor in other studies (Baeza-Yates et al. 2006; Caverlee and Liu 2007; Gyöngyi et al. 2004; Page et al. 1999; Wu et al. 2006; Zhang et al. 2011). In the PageRank-based functional ranking model (Baeza-Yates et al. 2006), the decay factor is an argument of the damping function affecting the speed of score attenuation.

In the proposed *USPM*, a page's *FS* (*BS*) consists of two parts [refer to Formulas (5) and (6)]: the first part receives the score from the score propagation process, and the second part receives the score from the distribution vector. The jump probability is used to adjust the weights of the two parts to determine the final score of a page. A smaller jump probability is highly trusted in the first part (score propagation) and less trusted in the distribution vector. It is reasonable to set a small jump probability since human experts can only evaluate very small web pages, and hence, the states (spam or normal) of most pages are unknown. In this paper, we set the jump probability to 0.15 as most previous models did for simplicity.

4 Proposed algorithms

In this section, we first review some published *WSDA*s and point out their defects. Then, two new *WSDA*s are introduced by taking advantage of the sub-functions proposed in the paper.

4.1 Review of WSDAs

On the basis of the *USPM*, a *WSDA* can be thought of as a decomposable system that consists of five components: splitting function, accepting function, combination function, initializer and jump probability, each of which implements two different versions for the forward score propagation function and backward score propagation function.

In this manner, we list some well-known *WSDA*s in Tables 1 and 2. The initialization function of the page state is ignored because all algorithms listed here use the same initialization method for the page state and distribution vector.

As we can see in Tables 1 and 2, the uniform splitting function is adopted in all algorithms except for GBR, which uses a proportional decay factor for uniform splitting. However, as discussed in Sect. 3, uniform splitting suffers from the problem that less trusted pages may propagate greater scores than trusted pages due to their very few outgoing links.

TDR is the first algorithm focusing on designing a differential propagation process by utilizing information contained in the target node. This yields a breakthrough in the definition of the score propagation function since the score can be propagated in a target differential manner. However, TDR has a drawback in that its accepting function largely limits the spread of scores. The accepting functions for *FSPF* and *BSPF* defined in TDR are

$$f^1_{accept}(p,fs) = \begin{cases} fs \times \dfrac{\beta \cdot FS(p)}{\beta \cdot FS(p) + (1-\beta) \cdot BS(p)}, & FS(p) \neq 0 \text{ or } BS(p) \neq 0 \\ fs, FS(p) = BS(p) = 0 \end{cases}, \quad 0 \leq \beta \leq 1, \tag{21}$$

$$f^2_{accept}(p,bs) = \begin{cases} bs \times \dfrac{(1-\beta) \cdot BS(p)}{\beta \cdot FS(p) + (1-\beta) \cdot BS(p)}, & FS(p) \neq 0 \text{ or } BS(p) \neq 0 \\ bs, FS(p) = BS(p) = 0 \end{cases}, \quad 0 \leq \beta \leq 1. \tag{22}$$

Pages that are not in the seed set are initialized with zero *FS* and *BS* in TDR. From Eqs. (21) and (22), it is obvious that a non-seed page with zero *FS* and zero *BS* can accept a non-zero *FS* or *BS*. However, if the page has a non-zero *BS* and a zero *FS*, it cannot accept the *FS* from its parents any longer due to the zero numerator in Eq. (21). Similarly, a page cannot accept the *BS* any longer if it has a non-zero *FS* and a zero *BS* due to the zero numerator in Eq. (22). Thus, a very large portion of web pages cannot receive any *FS* or *BS* in the score propagation process in TDR.

Different from TDR, which restricts the score propagation at the target end, GBR restricts the propagation process at the source end (Liu et al. 2013). As reported in (Liu et al. 2013), GBR can yield slightly better results than TDR. However, why can we not restrict score propagation at both the source and target ends? Indeed, it is truly very difficult to analyse these empirical models without high abstraction. In essence, TDR and GBR are quite different in that they focus on different aspects of the *USPM* (i.e., the splitting function and accepting function, respectively), even though they use very similar methods to penalize the base score.

Another interesting phenomenon is that all algorithms in Tables 1 and 2 are based on the simple summation combination function, which is vulnerable to deliberate manipulation. This again demonstrates the importance of the highly abstracted *USPM* because of its

Table 1 *FSPF* settings of *WSDA*s

Algorithms	$f^1_{split}(p)$	$f^1_{accept}(q, fs)$	$f^1_{com}(S)$	$dv_1(p)$
PageRank	$\frac{FS(p)}{\lvert Out(p)\rvert}$	fs	$\sum_{i \in S} i$	$\frac{1}{N}$
InversePageRank	–	–	–	–
TrustRank	$\frac{FS(p)}{\lvert Out(p)\rvert}$	fs	$\sum_{i \in S} i$	$\begin{cases} 1/\lvert SD_n\rvert, & p \in SD_n; \\ 0, & p \notin SD_n. \end{cases}$
Anti-TrustRank	–	–	–	–
LCRank	$\frac{FS(p)}{\lvert Out(p)\rvert}$	fs	$\sum_{i \in S} i$	$\begin{cases} 1/\lvert SD_n\rvert, & p \in SD_n; \\ 0, & p \notin SD_n. \end{cases}$
TDR	$\frac{FS(p)}{\lvert Out(p)\rvert}$	$\frac{fs \cdot \beta \cdot FS(q)}{\beta \cdot FS(q) + (1-\beta) \cdot BS(q)}$	$\sum_{i \in S} i$	$\begin{cases} 1/\lvert SD_n\rvert, & p \in SD_n; \\ 0, & p \notin SD_n. \end{cases}$
GBR	$\frac{FS(p)}{\lvert Out(p)\rvert} \times \frac{FS(p)}{FS(p)+BS(p)}$	fs	$\sum_{i \in S} i$	$\begin{cases} 1/\lvert SD_n\rvert, & p \in SD_n; \\ 0, & p \notin SD_n. \end{cases}$

Jump probability is set to 0.15. N: number of nodes in the web graph; SD_n: the set of non-spam seeds; –: the algorithm does not use *FSPF*; β: parameter used to adjust weights of *FS* and *BS* in functions, $0 \leq \beta \leq 1$

Table 2 *BSPF* settings of *WSDA*s. Jump probability is set to 0.15

Algorithms	$f^2_{split}(p)$	$f^2_{accept}(q, bs)$	$f^2_{com}(S)$	$dv_2(p)$
PageRank	–	–	–	–
InversePageRank	$\frac{BS(p)}{\lvert In(p)\rvert}$	bs	$\sum_{i\in S} i$	$\frac{1}{N}$
TrustRank	–	–	–	–
Anti-TrustRank	$\frac{BS(p)}{\lvert In(p)\rvert}$	bs	$\sum_{i\in S} i$	$\begin{cases} 1/\lvert SD_s\rvert, & p \in SD_s; \\ 0, & p \notin SD_s. \end{cases}$
LCRank	$\frac{BS(p)}{\lvert In(p)\rvert}$	bs	$\sum_{i\in S} i$	$\begin{cases} 1/\lvert SD_s\rvert, & p \in SD_s; \\ 0, & p \notin SD_s. \end{cases}$
TDR	$\frac{BS(p)}{\lvert In(p)\rvert}$	$\frac{bs\cdot(1-\beta)\cdot BS(q)}{\beta\cdot FS(q)+(1-\beta)\cdot BS(q)}$	$\sum_{i\in S} i$	$\begin{cases} 1/\lvert SD_s\rvert, & p \in SD_s; \\ 0, & p \notin SD_s. \end{cases}$
GBR	$\frac{BS(p)}{\lvert In(p)\rvert} \times \frac{BS(p)}{FS(p)+BS(p)}$	bs	$\sum_{i\in S} i$	$\begin{cases} 1/\lvert SD_s\rvert, & p \in SD_s; \\ 0, & p \notin SD_s. \end{cases}$

N: number of nodes in the web graph; SD_s: the set of spam seeds; –: the algorithm does not use *BSPF*; β: parameter used to adjust weights of *FS* and *BS* in functions, $0 \leq \beta \leq 1$

good interpretability. For example, following the mathematical form of PageRank like most algorithms do, we cannot easily define other mathematical forms of the score summation function because of the very specific meanings of the PageRank model.

Algorithms that propagate both trust and distrust (e.g., LCRank, TDR and GBR) adopt quite similar methods for the forward and backward score propagations. In other words, the mathematical definition of trust score and distrust score propagations of these algorithms are *symmetrical*, i.e., they use the same functions for *FSPF* and *BSPF*. This is open to argument. For example, in the web environment, a link between two web pages can be seen as an implication of trust from the source page to the target page in most cases. Thus, it is reasonable to say that trust can be propagated from a parent page to its children through links. However, score propagation for distrust is not as simple as trust because we cannot say that distrust is convened in the inverse link if we do not know the goodness of the target page (Krishnan and Raj 2006). Thus, it is more reasonable that *FS* and *BS* should be propagated in different manners. Unfortunately, symmetrical algorithms do not meet the requirement.

4.2 Our algorithms

To overcome the above-mentioned drawbacks of previous algorithms, we propose a new algorithm named *SFBR* (Supervised Forward and Backword Ranking) based on the sub-functions introduced in the paper.

As reported in (Liu et al. 2013; Zhang et al. 2011), TDR and GBR can achieve better results than PageRank, TrustRank and LCRank. This indicates the positive effect of the proportional decay factor on the propagation process, either at the source side or target side. Thus, we also adopt the proportional decay factor in the splitting functions of *SFBR*. The forward splitting function of *SFBR* is defined as

$$f^1_{split}(p) = \begin{cases} Base_1(p) \times \dfrac{\beta \cdot FS(p)}{\beta \cdot FS(p) + (1-\beta) \cdot BS(p)}, & FS(p) \neq 0 \\ Base_1(p), FS(p) \end{cases}, \quad 0 \leq \beta \leq 1, \quad (23)$$

and the backward splitting function is

$$f^2_{split}(p) = \begin{cases} Base_2(p) \times \dfrac{(1-\beta) \cdot BS(p)}{\beta \cdot FS(p) + (1-\beta) \cdot BS(p)}, & BS(p) \neq 0 \\ Base_2(p), BS(p) = 0 \end{cases}, \quad 0 \leq \beta \leq 1. \quad (24)$$

To overcome the drawbacks of the uniform splitting base score function used in TDR and GBR, we adopt the logarithm splitting for forward and backward splitting functions in *SFBR*. It is reasonable to model the behaviour of the score splitting process such that scores propagated from a source page should have a negative relation with its number of outgoing links since the possibility of a web page linking to spam increases with the number of links that it has increases (Wu et al. 2006). However, there is no obvious clue that the negative relation between the propagated score and the number of outgoing links is linear (as in TDR and GBR). We investigate the proportion of outgoing spam pages of top-10 ranked normal pages in the WEBSPAM-UK2007 (Yahoo! 2007) dataset using the PageRank algorithm. The results show that of 169 labelled outgoing links of the top ranked 10 normal pages, only one of them is spam. Making use of logarithm splitting can decrease the effect of the number of outgoing links, as well as keep the propagated score closer to the source page's forward score. Thus, we have the forward and backward splitting functions of the *SFBR* algorithm:

$$f^1_{split}(p) = \begin{cases} \dfrac{FS(p)}{\log(1+|Out(p)|)} \times \dfrac{\beta \cdot FS(p)}{\beta \cdot FS(p) + (1-\beta) \cdot BS(p)}, & FS(p) \neq 0 \\ \dfrac{FS(p)}{\log(1+|Out(p)|)}, & FS(p) = 0 \end{cases}, \quad 0 \leq \beta \leq 1, \quad (25)$$

and

$$f^2_{split}(p) = \begin{cases} \dfrac{BS(p)}{\log(1+|In(p)|)} \times \dfrac{(1-\beta) \cdot BS(p)}{\beta \cdot FS(p) + (1-\beta) \cdot BS(p)}, & BS(p) \neq 0 \\ \dfrac{BS(p)}{\log(1+|In(p)|)}, & BS(p) = 0 \end{cases}, \quad 0 \leq \beta \leq 1. \quad (26)$$

In the *USPM*, the score propagation is modelled by three sub-functions. Intuitively, we expect a *WSDA* to achieve better performance if the sub-functions defined in the *WSDA* make the model more coincident with the real case. However, it is difficult to accurately capture every aspect of the real case, so we focus on the most important aspects of the model as most previous algorithms have done, e.g., the approximate isolation principle for good sites and bad sites.

The success of the PageRank algorithm has demonstrated some valuable merits of the model in the propagation of *FS*. For example, the simple summation function plays a significant role in the performance of quantity and quality examinations of a page's parents, even though it easily suffers from deliberate human manipulation. However, it violates the principle of designing a *WSDA* such that we adopt other summation functions to

eliminate the effect caused by deliberate manipulation, such as link boosting, since such link structures are very common in real web environments among normal pages and eliminating those links would also result in a huge negative effect on many normal pages. With this in mind, we adopt the same accepting and combination functions as PageRank for *FS*:

$$f^1_{accept}(p,fs) = fs, \tag{27}$$

$$f^1_{combination}(S) = \sum_{e \in S} e. \tag{28}$$

The constant accepting and simple combination functions defined in Eqs. (27) and (28) are inherently suitable for *FS* propagation since citation count is a very important measure in the real web (Page et al. 1999). However, this is not true for *BS* propagation. The approximate isolation principle for bad sites notes that it is very unlikely for spam pages to be pointed to by good pages, and thus, distrust can be propagated through inverse links from the bad source sites. However, it is meaningless to say that the inverse links convey distrust if we do not know the goodness or badness of the source site. Thus, the inverse citation count cannot be directly used as a measure for spam. A very important issue in the design of the propagation of *BS* is to ensure that a normal page would not receive too high *BS* even if it points to some suspected pages due to careless, which probability can likely be roughly evaluated by its number of outgoing links (Gyöngyi et al. 2004). Therefore, a page's accepted *BS* is penalized if it has many outgoing links:

$$f^2_{accept}(p, bs) = \frac{bs}{|Out(p)|}. \tag{29}$$

The top-n score combination function is used to further penalize *BS* for pages with many outgoing links. With $n = \lfloor \log(1 + |Out(p)|) \rfloor$,

$$f^2_{combination}(S) = \sum_{i=1}^{\lfloor \log(1+|Out(p)|) \rfloor} S_{sort}(i), \tag{30}$$

where p is the current page, S_{sort} denotes the result set by sorting S in descending order and $S_{sort}(i)$ represents the i-th largest element in S_{sort}. This combination function accumulates the most dis-trusted scores obtained from source pages. Thus, a page gets high *BS* if it points to some pages with high *BS*s.

SFBR is asymmetrical since different sub-functions are used in *FSPF* and *BSPF*. To the best of our knowledge, *SFBR* is the first asymmetrical algorithm designed to demote spam. We simply obtain an unsupervised version of *SFBR* by eliminating the effects of good and bad seeds and initialize the *FS*, *BS* and distribution vector value of every node to $1/N$, where N is the number of sites in the web graph. This unsupervised algorithm is named Unsupervised Forward and Backward Ranking (*UFBR*). Tables 3 and 4 present an intuitive view of *SFBR* and *UFBR*.

Table 3 *FSPF* settings of *SFBR* and *UFBR*

Algorithms	$f^1_{split}(p)$	$f^1_{accept}(q, fs)$	$f^1_{com}(S)$	$dv_1(p)$
SFBR	$\frac{FS(p)}{\log(1+\lvert Out(p)\rvert)} \times \frac{\beta \cdot FS(p)}{\beta \cdot FS(p)+(1-\beta)\cdot BS(p)}$	fs	$\sum_{i \in S} i$	$\begin{cases} 1/SD_n, & p \in SD_n \\ 0, & p \notin SD_n \end{cases}$
UFBR	$\frac{FS(p)}{\log(1+\lvert Out(p)\rvert)} \times \frac{\beta \cdot FS(p)}{\beta \cdot FS(p)+(1-\beta)\cdot BS(p)}$	fs	$\sum_{i \in S} i$	$\frac{1}{N}$

Jump probability is set to 0.15. N: number of nodes in the web graph; SD_n: the set of non-spam seeds; β: parameter used to adjust weights of FS and BS in functions, $0 \leq \beta \leq 1$

Table 4 *BSPF* settings of *SFBR* and *UFBR*

Algorithms	$f^2_{split}(p)$	$f^2_{accept}(q, bs)$	$f^2_{com}(S)$	$dv_2(p)$
SFBR	$\frac{BS(p)}{\log(1+\lvert In(p)\rvert)} \times \frac{(1-\beta)\cdot BS(p)}{\beta \cdot FS(p)+(1-\beta)\cdot BS(p)}$	$\frac{bs}{\lvert Out(p)\rvert}$	$\sum_{i=1}^{\lfloor \log(1+\lvert Out(p)\rvert)\rfloor} S_{sort}(i)$	$\begin{cases} 1/\lvert SD_s\rvert, & p \in SD_s \\ 0, & p \notin SD_s \end{cases}$
UFBR	$\frac{BS(p)}{\log(1+\lvert In(p)\rvert)} \times \frac{(1-\beta)\cdot BS(p)}{\beta \cdot FS(p)+(1-\beta)\cdot BS(p)}$	$\frac{bs}{\lvert Out(p)\rvert}$	$\sum_{i=1}^{\lfloor \log(1+\lvert Out(p)\rvert)\rfloor} S_{sort}(i)$	$\frac{1}{N}$

Jump probability is set to 0.15. N: number of nodes in the web graph; SD_s: the set of spam seeds; β: parameter used to adjust weights of FS and BS in functions, $0 \leq \beta \leq 1$

5 Experiments and discussions

5.1 Datasets

We conducted experiments on three large public datasets: WEBSPAM-UK2006 (Castillo et al. 2006), WEBSPAM-UK2007 (Yahoo! 2007) and EU2010 (András et al. 2010). The first dataset is used for the web spam challenge 2007 (Cormack 2007) and contains 77 M pages from 11,402 hosts crawled from the UK domain. The base data of WEBSPAM-UK2007 is a set of 100 M pages in 114,529 hosts. The data were downloaded in May 2007 and used for the web spam challenge 2008 (Castillo et al. 2008). Some hosts in the WEBSPAM-UK2007 dataset are labelled as undecided, and we removed them from the labelled host set in our experiment since they are useless for the web spam demotion study. EU2010 was created for the ECML/PKDD Discovery Challenge 2010 on Web Quality and is a large collection of annotated Web hosts labelled by the Hungarian Academy of Sciences (English), European Archive Foundation (French) and L3S Hannover (German). The base data of EU2010 is a set of 23 M pages in the EU domain (András et al. 2010).

For convenience, all isolation nodes (no incoming and outgoing links) are removed from the graph since they cannot contribute to the score propagation process. Some statistics of these three datasets are shown in Table 5.

5.2 Evaluation metrics

In the proposed *USPM*, the state of a web page is defined by a pair of values, i.e., the page's *FS* and *BS*. We formalize metrics for evaluating the ability of a *WSDA* to demote and detect spam based on *FS*s and *BS*s, respectively.

Table 5 Statistics of WEBSPAM-UK2006/2007 and EU2010 datasets

Dataset	Hosts	Labeled hosts	Normal hosts	Spam hosts	Spam ratio (%)	Isolation spam	Isolation normal
WEBSPAM-UK2006	11,402	7473	5549	1924	26	4	0
WEBSPAM-UK2007	114,529	6053	5709	344	6	24	0
EU2010	191,388	1966	1897	69	3.5	10	0

5.2.1 Top-k spam factor

*FS*s can be used to rank web pages for search engine applications in a similar manner to the PageRank scores because *FS* is designed to characterize the good side of a web page in the *USPM*. Pages are reordered by their *FS*s, and the pages with higher rankings are expected to have a higher likelihood of being good than those with lower rankings. The top-k spam factor metric is defined to evaluate the ability of a *WSDA* to demote web spam by examining the number of spam sites and their rankings in the list ordered by *FS*s in descending order.

Given a ranked list L_f of n sites $s_1, s_2, \ldots, s_n$ that is sorted in descending order by *FS*s, the top-k spam factor (*TKSF*) of L_f is

$$TKSF(L_f, k) = \frac{\sum_{i=1}^{k} w(s_i) \times \frac{1}{i}}{\sum_{i=1}^{k} \frac{1}{i}}, \quad where\, 0 < k \leq n\, and\, w(s_i) = \begin{cases} 1, & if\ s_i\ is\ spam \\ 0, & if\ s_i\ is\ normal \end{cases}. \tag{31}$$

TKSF increases if more spam sites that have high rankings are ranked in the top-k results. *TKSF* equals 0 if there is no spam site in the top-k results and equals 1 if all of them are spam. The spam factor used in (Liu et al. 2013) is a specific case of *TKSF* by setting k to n, in which all spam sites will be ranked in the top-k results, and so only rankings of them affect the final result. It is essential for search engines to demote spam in top-ranked sites since most users are only interested in them. We use different k to evaluate the ability of a *WSDA* to eliminate spam sites from top positions.

5.2.2 Top-k spam precision

BS describes the badness of a web page and can be used to detect spam in a simple manner. We can choose a threshold t and predict a page as spam if its *BS* is larger than t. Alternately, in another approach, we can predict pages with the highest rankings as spam in the list reordered by *BS*s in descending order. We use the later method to evaluate the ability of a *WSDA* to detect spam by their pages' *BS*s since the score computed in the *WSDA* is a local relative value, and hence, we cannot find a reasonable threshold t to evaluate the performance of a *WSDA* on different datasets. For example, if page A and page B are in different datasets, it is almost meaningless to say that page A is more "important" than page B even if A has a higher PageRank score than B.

Given a ranked list L_b of n sites $s_1, s_2, \ldots, s_n$ that is sorted in descending order by *BS*s, the top-k spam precision (*TKSP*) of L_b is

$$TKSP(L_b,k)=\frac{\sum_{i=1}^{k} w(s_i)}{k},\quad where\ 0<k\le n\ and\ w(s_i)=\begin{cases}1, & if\ s_i\ is\ spam\\ 0, & if\ s_i\ is\ normal\end{cases}. \tag{32}$$

The *TKSP* value is affected only by the number of spam sites, not by their positions, and it increases with more spam sites ranked in the top-k results of the list L_b.

5.3 Experimental results

Experiments are conducted on three publically available datasets, WEBSPAM-UK2006 (Castillo et al. 2006), WEBSPAM-UK2007 (Yahoo! 2007) and EU2010 (András et al. 2010). All algorithms, including previous schemes (Gyöngyi et al. 2004; Krishnan and Raj 2006; Liu et al. 2013; Page et al. 1999; Wu et al. 2006; Zhang et al. 2011) and the proposed *SFBR*, use the same set of seeds selected by PageRank (for good seeds) and Inverse-PageRank (for bad seeds) for comparison. Experiments are conducted regarding the effect of the seed set size, which varies according to the data distribution of different datasets. For WEBSPAM-UK2006 and WEBSPAM-UK2007, the seed set consists of the same number of good and bad sites, and its size is changed from 40 to 160, with an increment of 40 per experiment. For the EU2010 dataset, the seed set is fixed at 40 spam pages and 100 normal pages since there are very few labelled spam samples and many good sites are in small connected components. Thus, we need a larger good seed set to ensure propagation of *FS*s.

5.3.1 Spam demotion

The performance of an algorithm for spam demotion is evaluated through the top-k spam factor (*TKSF*). The parameter k of *TKSF* is changed from 50 to the largest size of the dataset, with an increment of 50 per experiment, to test the performance of the algorithm on different sizes of top ranked results. For simplicity, we only present the most meaningful experimental data in the paper, instead of providing all of them, due to the large amount of data.

The experimental results are shown in Tables 6, 7 and 8, in which the best performance of each row is highlighted in bold. The "k" used in these tables varies according to results on datasets. For example, the *TKSF*s of all algorithms except PageRank are 0 in the top 1650 results on the WEBSPAM-UK2006 dataset. Thus, we do not provide those data because of their low value for algorithm comparison. The global views of the results are drawn in Figs. 2, 3 and 4.

It is shown that the proposed *SFBR* always outperforms the others on the three datasets. The results demonstrate the robustness and effectiveness of *SFBR* in demoting web spam. A very interesting phenomenon in Figs. 2, 3 and 4 is that the shapes of the result curves of all previous supervised algorithms (Gyöngyi et al. 2004; Liu et al. 2013; Wu et al. 2006; Zhang et al. 2011) are very similar. Referring to Tables 1 and 2, we can see that those algorithms are largely based on the uniform splitting, constant accepting and simple summation combination. The disadvantages of the base components used in those algorithms limit their performance. *SFBR* breaks the limitations by taking advantage of different sub-functions for *FSPF* and *BSPF* depending on their different behaviours of score propagation. For example, *SFBR* uses a constant accepting function and a simple summation function for *FSPF* since the number of citations is indeed a very important indicator for good pages, and hence, *SFBR* can benefit from constant accepting and simple summation to assign high *FS*s to popular pages. However, the case for *BSPF* is quite

Table 6 *TKSF*s of algorithms on WEBSPAM-UK2006

Seed set size	k	PageRank	TrustRank	LCRank	TDR	GBR	SFBR	UFBR
40	1700	0.04985	2.64E−04	3.56E−04	2.62E−04	3.49E−04	**0**	9.46E−05
	1800	0.05090	3.41E−04	3.53E−04	3.42E−04	3.46E−04	**7.98E−05**	9.39E−05
	1900	0.05128	4.16E−04	5.01E−04	4.16E−04	4.21E−04	**7.92E−05**	9.32E−05
80	1700	0.04985	4.67E−04	5.63E−04	4.56E−04	4.71E−04	**0**	9.46E−05
	1800	0.05090	5.42E−04	5.59E−04	5.32E−04	5.48E−04	**8.01E−05**	9.39E−05
	1900	0.05128	5.38E−04	7.05E−04	5.28E−04	5.44E−04	**7.95E−05**	9.32E−05
120	1700	0.04985	4.46E−04	5.46E−04	5.24E−04	5.38E−04	**0**	9.46E−05
	1800	0.05090	5.26E−04	5.42E−04	5.20E−04	5.34E−04	**7.99E−05**	9.39E−05
	1900	0.05128	5.22E−04	6.87E−04	5.16E−04	5.30E−04	**7.93E−05**	9.32E−05
160	1700	0.04985	4.67E−04	3.96E−04	4.56E−04	4.67E−04	**0**	9.46E−05
	1800	0.05090	4.63E−04	4.75E−04	4.53E−04	4.63E−04	**7.97E−05**	9.39E−05
	1900	0.05128	4.60E−04	4.72E−04	4.49E−04	4.60E−04	**7.91E−05**	9.32E−05

The best performance of each row is highlighted in bold

Table 7 *TKSF*s of algorithms on WEBSPAM-UK2007

Seed set size	k	PageRank	TrustRank	LCRank	TDR	GBR	SFBR	UFBR
40	100	0.013441	0.003094	0.003220	0.002836	0.005177	**0**	0
	200	0.014253	0.005368	0.005873	0.005708	0.005923	**0.002259**	0.005236
	300	0.014804	0.005673	0.007581	0.005273	0.005472	**0.002730**	0.005570
80	100	0.013441	0.003404	0.003610	0.003054	0.002978	**0**	0
	200	0.014253	0.003987	0.004337	0.003733	0.003704	**0.001365**	0.005236
	300	0.014804	0.004371	0.005432	0.004312	0.004344	**0.002802**	0.005570
120	100	0.013441	0.003094	0	0	0	**0**	0
	200	0.014253	0.002660	0.001151	0.001072	0.002601	**0**	0.005236
	300	0.014804	0.003389	0.002382	0.001691	0.003145	**0.001529**	0.005570
160	100	0.013441	0.002647	0	0	0	**0**	0
	200	0.014253	0.002275	0.002130	0.001119	0.002507	**0**	0.005236
	300	0.014804	0.003876	0.001968	0.001922	0.003179	**0.001511**	0.005570

The best performance of each row is highlighted in bold

Table 8 *TKSF*s of algorithms on EU2010

Seed set size	k	PageRank	TrustRank	LCRank	TDR	GBR	SFBR	UFBR
100 normal 40 spam	150	0.011959	0.001521	0.001521	0.001532	0.001532	**0.001490**	0.019416
	250	0.013850	0.002345	0.001370	0.001380	0.001380	**0.001342**	0.020383
	350	0.013594	0.003476	0.001286	0.001295	0.002487	**0.001259**	0.023949

The best performance of each row is highlighted in bold

different because inverse-citations are not explicit indications of bad pages. Therefore, *SFBR* adopts some proposed sub-functions to keep good pages from receiving backward scores as much as possible.

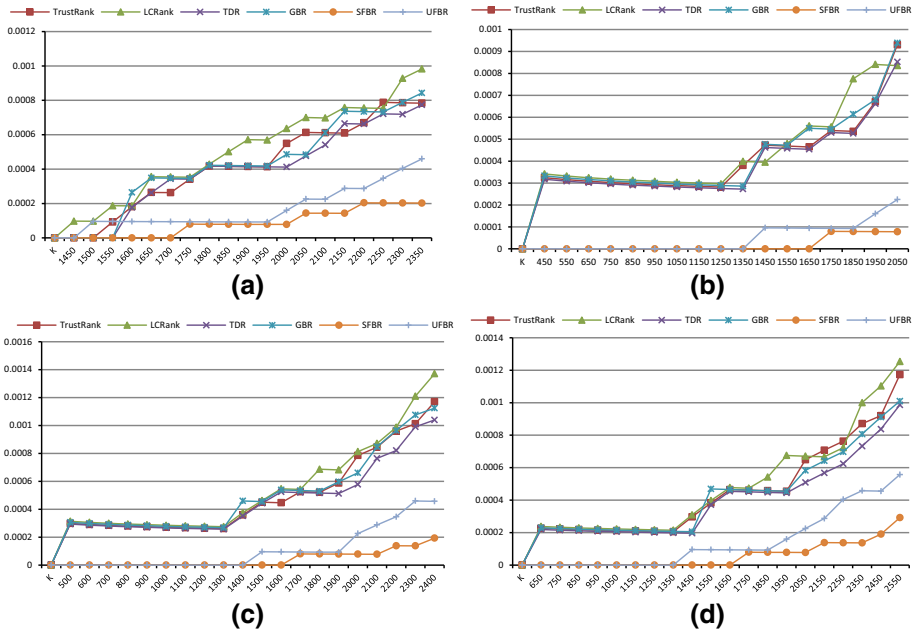

Fig. 2 *TKSF*s of 6 algorithms on WEBSPAM-UK2006. X-axis: *K*; Y-axis: Top-*k* spam factor. The PageRank algorithm is removed from the figure since its performance is much worse than others. Adding it into the figure leads to very poor visualization of data. **a** *TKSF*s on WEBSPAM-UK2006 with seed size 40. **b** *TKSF*s on WEBSPAM-UK2006 with seed size 80. **c** *TKSF*s on WEBSPAM-UK2006 with seed size 120. **d** *TKSF*s on WEBSPAM-UK2006 with seed size 160

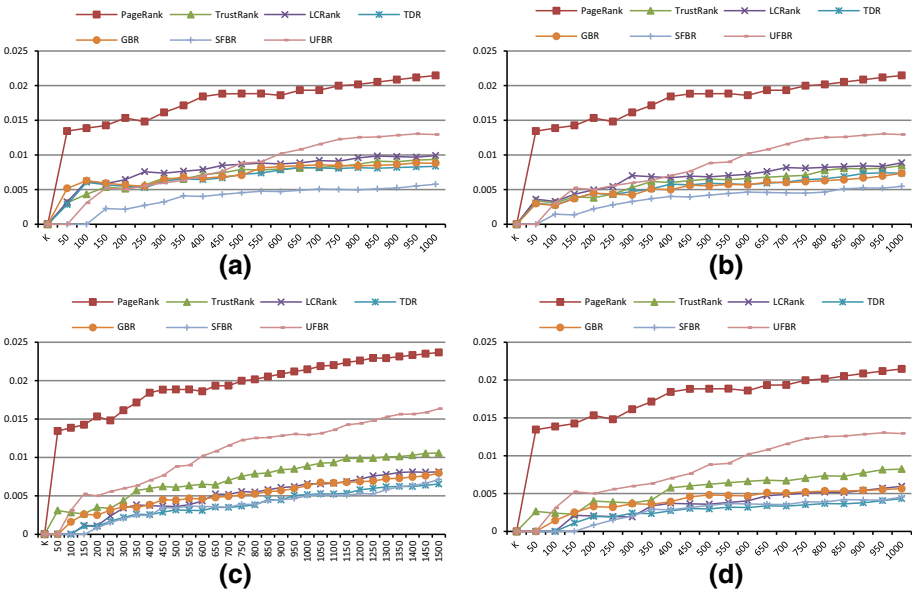

Fig. 3 *TKSF*s of 7 algorithms on WEBSPAM-UK2007. X-axis: *K*; Y-axis: Top-*k* spam factor. **a** *TKSF*s on WEBSPAM-UK2007 with seed size 40. **b** *TKSF*s on WEBSPAM-UK2007 with seed size 80. **c** *TKSF*s on WEBSPAM-UK2007 with seed size 120. **d** *TKSF*s on WEBSPAM-UK2007 with seed size 160

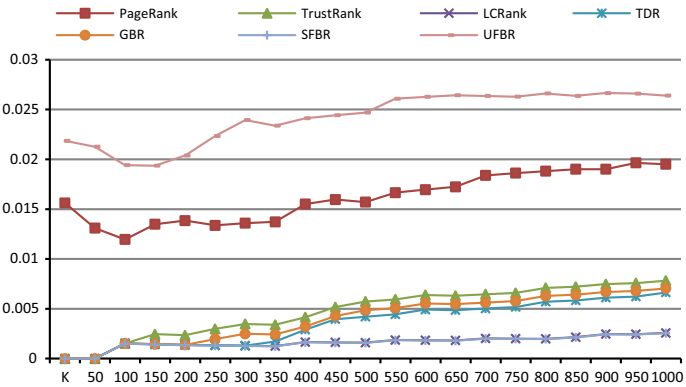

Fig. 4 *TKSF*s of 7 algorithms on EU2010 with seed set consisting of 40 spam sites and 100 normal sites. X-axis: *K*; Y-axis: Top-*k* spam factor

Table 9 Top-10 ranked spam site lists of TrustRank and UFBR

Number	TrustRank		UFBR	
	Site ID	Ranking position	Site ID	Ranking position
1	9863	1558	10980	1506
2	6372	1603	6078	2047
3	10980	1700	7307	2091
4	7668	1781	7047	2163
5	4707	1814	6072	2276
6	3447	2016	4707	2315
7	6078	2033	5890	2396
8	6072	2093	3999	2577
9	9161	2205	8328	2586
10	4159	2253	3880	2668

Another important clue for proving the effectiveness of *SFBR* is the very promising result obtained by its unsupervised version, *UFBR*. Figure 2 clearly shows that *UFBR* obtains better results than the previous algorithms on WEBSPAM-UK2006, even though *UFBR* is unsupervised. This indicates that the functions used in *SFBR* and *UFBR* are better approximations of the real underlying score propagation behaviours. On WEBSPAM-UK2007, *UFBR* still outperforms the others besides *SFBR* in the top-100 results for different sizes of the seed set. The result is encouraging since it shows the possibility of designing unsupervised *WSDA*s that can yield comparable results to supervised algorithms by constructing reasonable propagation functions.

We present some detail data of the experiment on WEBSPAM-UK2006 dataset to illustrate how the unsupervised algorithm *UFBR* can achieve better results than supervised algorithms. The top-10 ranked spam sites of TrustRank and *UFBR* are listed in Table 9 as an example.

As shown in Table 9, there are 4 sites (10980, 4707, 6078 and 6072) contained in both lists and only the site 10980 has higher ranking position in *UFBR* than in TrustRank.

TrustRank propagates *FS* scores from a set of seeds. In an iteration of score propagation processing, the *FS* propagated from a seed moves one distance forward, as shown in Fig. 5.

The *FS* contribution of the seed p_1 to node decreases exponentially fast as the distance of the node to p_1 increases. Therefore, sites that are near seeds have much higher probability of being assigned higher forward scores than sites that are far from seeds. We computed the shortest distances to seeds for sites in the list of TrustRank and found that all of them are smaller than 5. That means TrustRank is vulnerable to good-to-bad links, and hence spam sites that can be accessed from seed sites in few steps could get high forward scores. For example, the site 3447 contains only 3 in-links, but it is ranked higher than many normal sites by TrustRank just because of its short distance to the seeds (the shortest distance is 4).

The unsupervised algorithm *UFBR* uses heuristics to guide the algorithm to demote spam because of the lack of prior human labeled knowledge. The in-link structure is used

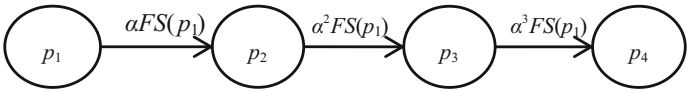

Fig. 5 Example of score propagation processing of TrustRank. p_1 is a seed site and its *FS* is $FS(p_1)$. In the first iteration, p_1 propagates its decayed forward score $\alpha FS(p_1)$ to its child p_2, where α is the jump probability. Because the initial *FS* scores and distribution probabilities for p_2, p_3 and p_4 are zeros, p_3 and p_4 cannot get any *FS* scores in the first iteration. In the second iteration, p_3 gets forward score $\alpha FS(p_2)$ from p_2. By substituting $FS(p_2)$, p_3 obtains $\alpha^2 FS(p_1)$ forward score. Similarly, p_4 will get $\alpha^3 FS(p_1)$ forward score in the third iteration

Table 10 *TKSP*s of algorithms on WEBSPAM-UK2006

Seed set size	k	InversePageRank	AntiTrustRank	LCRank	TDR	GBR	SFBR	UFBR
40	50	0.9000	**0.9200**	**0.9200**	**0.9200**	**0.9200**	**0.9200**	0.7800
	100	0.8900	0.9300	0.9300	0.9300	0.9300	**0.9400**	0.5800
	200	0.8100	0.8900	0.8950	0.8900	0.8900	**0.9050**	0.5250
	350	0.5660	0.7114	0.7828	0.7857	0.7485	**0.8371**	0.4771
	550	0.4180	0.5036	0.5909	0.6418	0.5363	**0.7454**	0.4636
80	50	0.9000	0.9200	0.9200	0.9200	0.9200	**1.0000**	0.7800
	100	0.8900	0.9300	0.9300	0.9300	0.9300	**0.9400**	0.5800
	200	0.8100	0.8950	0.9000	0.9000	0.9000	**0.9250**	0.5250
	350	0.5660	0.7285	0.7857	0.8142	0.7600	**0.8542**	0.4771
	550	0.4180	0.5127	0.6000	0.6381	0.5527	**0.7345**	0.4636
120	50	0.9000	0.9400	0.9400	0.9400	0.9400	**0.9800**	0.7800
	100	0.8900	0.9300	0.9300	0.9300	0.9300	**0.9500**	0.5800
	200	0.8100	0.8950	0.9000	0.9050	0.8950	**0.9400**	0.5250
	350	0.5660	0.7371	0.7942	0.8257	0.7600	**0.8514**	0.4771
	550	0.4180	0.5145	0.6036	0.6545	0.5563	**0.7400**	0.4636
160	50	0.9000	0.9200	0.9200	**0.9400**	**0.9400**	**0.9400**	0.7800
	100	0.8900	0.9300	0.9300	0.9400	0.9400	**0.9700**	0.5800
	200	0.8100	0.8950	0.9000	0.9100	0.8950	**0.9250**	0.5250
	350	0.5660	0.7371	0.7942	0.8600	0.7628	**0.8542**	0.4771
	550	0.4180	0.5181	0.6254	0.6800	0.5727	**0.7290**	0.4636

The best performance of each row is highlighted in bold

Table 11 *TKSP*s of algorithms on WEBSPAM-UK2007

Seed set size	k	InversePageRank	AntiTrustRank	LCRank	TDR	GBR	SFBR	UFBR
40	50	0.1800	0.4400	0.4400	0.4400	0.4400	**0.5400**	0.0200
	100	0.1700	0.2900	0.3000	**0.3200**	0.3000	**0.3200**	0.0200
	200	0.1200	0.2000	0.2050	**0.2300**	0.2050	0.2050	0.0450
	500	0.0860	0.1060	0.1160	**0.1360**	0.1140	0.1260	0.0380
	1000	0.0690	0.0750	0.0810	0.1000	0.0780	**0.0830**	0.0510
80	50	0.1800	0.8000	0.8000	0.8000	0.8000	**0.8200**	0.0200
	100	0.1700	0.4300	0.4400	0.4400	0.4400	**0.4800**	0.0200
	200	0.1200	0.2300	0.2350	0.2550	0.2350	**0.2750**	0.0450
	500	0.0860	0.1100	0.1220	0.1400	0.1180	**0.1480**	0.0380
	1000	0.0690	0.0790	0.0830	0.0930	0.0830	**0.0950**	0.0510
120	50	0.1800	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.0200
	100	0.1700	0.6000	0.6000	0.6000	0.6000	**0.6100**	0.0200
	200	0.1200	0.3100	0.3100	0.3300	0.3100	**0.3400**	0.0450
	500	0.0860	0.1320	0.1360	0.1560	0.1360	**0.1600**	0.0380
	1000	0.0690	0.0830	0.0860	**0.0980**	0.0870	0.0930	0.0510
160	50	0.1800	0.9800	0.9800	**1.0000**	0.9800	**1.0000**	0.0200
	100	0.1700	**0.8000**	**0.8000**	**0.8000**	**0.8000**	**0.8000**	0.0200
	200	0.1200	0.4000	0.4050	0.4050	0.4050	**0.4200**	0.0450
	500	0.0860	0.1509	0.1509	0.1600	0.1509	**0.1690**	0.0418
	1000	0.0690	0.0890	0.0910	0.1010	0.0920	**0.0990**	0.0510

The best performance of each row is highlighted in bold

Table 12 *TKSP*s of algorithms on EU2010

Seed set size	k	InversePageRank	AntiTrustRank	LCRank	TDR	GBR	SFBR	UFBR
100 normal + 40 spam	50	0.0000	**0.8400**	**0.8400**	**0.8400**	**0.8400**	**0.8400**	0.0200
	100	0.0100	**0.4200**	**0.4200**	**0.4200**	**0.4200**	**0.4200**	0.0200
	200	0.0250	0.2100	**0.2150**	**0.2150**	0.2100	**0.2150**	0.0200
	300	0.0233	0.1400	**0.1566**	0.1433	0.1433	**0.1566**	0.0266
	400	0.0175	0.1125	**0.1200**	0.1075	0.1150	**0.1200**	0.0275

The best performance of each row is highlighted in bold

to evaluate the goodness of a node and the out-link structure is used to evaluate the badness of a node. *UFBR* can get good result if the defined heuristics match the underlying data distribution. We point out one obvious clue that the heuristics used in *UFBR* match the underlying data distribution of WEBSPAM-UK2006 dataset in some extent. The experiment result of InversePageRank in Table 10 shows that a site has high probability of being spam if it plays a very important role in the inverse-link structure. So using a badness score computed based on inverse-link structure to penalize forward score can demote many spam sites in the dataset. For example, the top ranked spam site (ID: 9863) by TrustRank is ranked 4958 by *UFBR* due to its too many outgoing links (more than 100). However, if a spam site has many in-links but few out-links, *UFBR* fails to demote it, such as the top ranked spam site 10980.

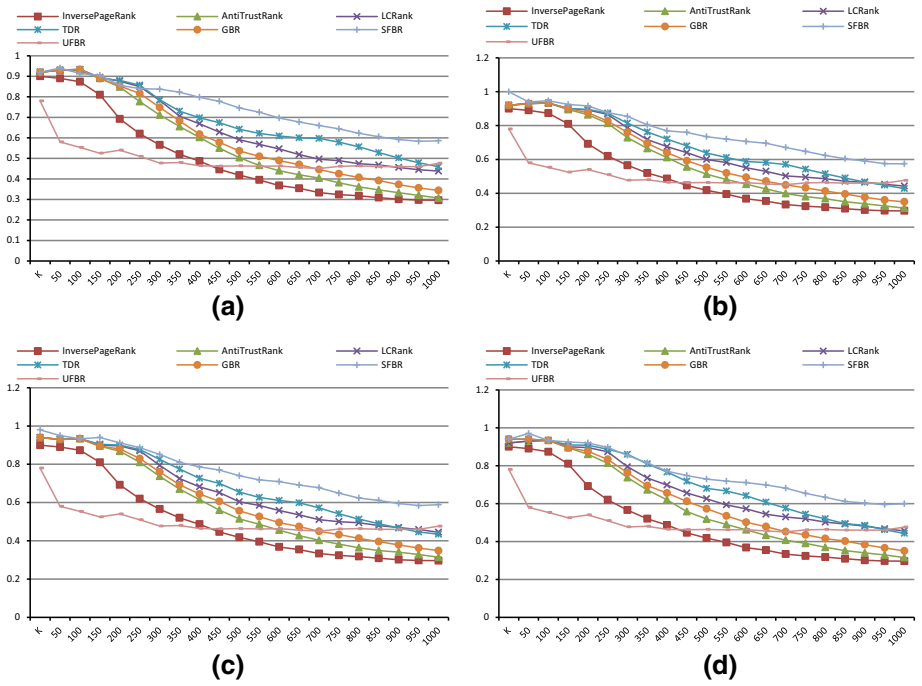

Fig. 6 *TKSP*s of 7 algorithms on WEBSPAM-UK2006. X-axis: *K*; Y-axis: Top-*k* spam precision. **a** *TKSP*s on WEBSPAM-UK2006 with seed size 40. **b** *TKSP*s on WEBSPAM-UK2006 with seed size 80. **c** *TKSP*s on WEBSPAM-UK2006 with seed size 120. **d** *TKSP*s on WEBSPAM-UK2006 with seed size 160

We found that the performance of *UFBR* drops largely on the EU2010 dataset. We further investigated the issue and found that it is related to the data distribution of the dataset. Different from WEBSPAM-UK2006 and WEBSPAM-UK2007, which are composed of a few large connected components, EU2010 consists of many small connected components, which leads to the difficulty of score propagation. For supervised algorithms, the performance is largely related to the seed set in such cases because pages of most components cannot obtain any *FS* and *BS* if there is no seed selected in them. Unfortunately, the seed select methods PageRank and InversePageRank are inherently biased to large connected components. Thus, on the EU2010 dataset, supervised algorithms can only propagate scores from the seed set to connected components that contain seed pages. However, all pages can still obtain *FS* and *BS* in unsupervised algorithms due to the uniform distribution vector. Thus, the score propagation process of unsupervised algorithms can be thought of as conducting the same algorithms on different graphs (each connected component is a graph) while still ranking these scores in the same list. Fortunately, this issue would not cause any problem of comparison for supervised algorithms, as they use the same seed set, and the page scores can be propagated to still be the same.

5.3.2 Spam detection

The performance of our proposed algorithm for spam detection is evaluated through top-*k* spam precision (*TKSP*). Adopting the same strategy used for *TKSF*, the parameter *k* of

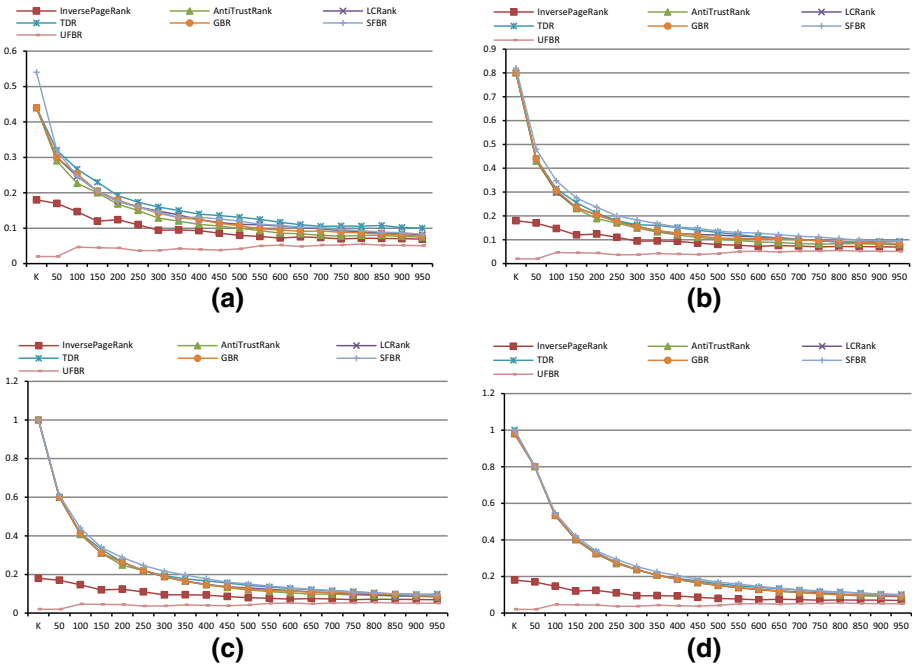

Fig. 7 *TKSP*s of 7 algorithms on WEBSPAM-UK2007. X-axis: *K*; Y-axis: Top-*k* spam precision. **a** *TKSP*s on WEBSPAM-UK2007 with seed size 40. **b** *TKSP*s on WEBSPAM-UK2007 with seed size 80. **c** *TKSP*s on WEBSPAM-UK2007 with seed size 120. **d** *TKSP*s on WEBSPAM-UK2007 with seed size 160

TKSP is changed from 50 to the largest size of the dataset with an increment of 50 per test. The *TKSP*s of algorithms are shown in Tables 10, 11 and 12. The global views of the results are drawn in Figs. 6, 7 and 8.

In summary, *SFBR* has the best performance among all algorithms in spam detection. From the results, we empirically provide the following conclusions:

(1) The constant accepting and top-*n* score summation can keep good pages from receiving backward scores as much as possible. Thus, the spam precision of *SFBR* is almost the largest.
(2) As the result of Inverse-PageRank shows, the number of outgoing links may be an indication of spam in some cases, e.g., in WEBSPAM-UK2006. This assumption needs further study since the WEBSPAM-UK2006 dataset has the most labelled spam sites, and hence, their distribution may be more uniform than that in the other two datasets.
(3) The bad performance of *UFBR* is not supervising since its design principle is to keep good pages from receiving backward scores instead of encouraging the *BS* to be propagated to other suspected pages. The reason is that the heuristic estimations of a suspected page cannot be easily defined, while this is much easier (or, in other words, more accurate) for normal pages.

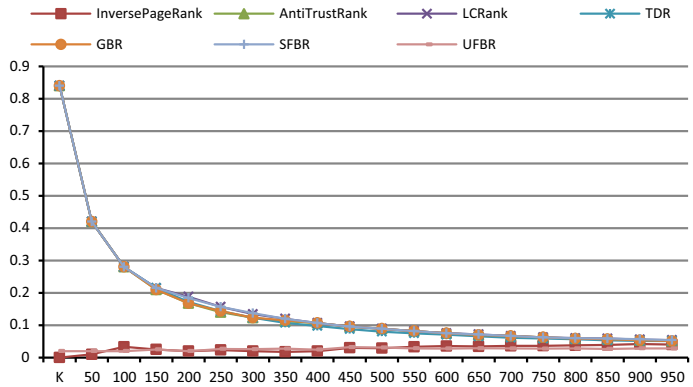

Fig. 8 *TKSF*s of 7 algorithms on EU2010. X-axis: *K*; Y-axis: Top-*k* spam factor. *TKSP*s on EU2010 with seed set consisting of 40 spam sites and 100 normal sites

6 Conclusions

In this paper, we proposed a unified score propagation model (*USPM*) for web spam demotion algorithms (*WSDAs*). A forward score propagation function (*FSPF*) and a backward score propagation function (*BSPF*) are introduced to model score propagation processes through links. *FSPF* and *BSPF* can be further expressed as three sub-functions: a splitting function, an accepting function and a combination function. Under the *USPM*, different algorithms can be simply thought of as scoring systems using different sub-functions. A set of candidate sub-functions, including previous methods and some newly introduced in this paper, are discussed. Taking advantage of the *USPM*, we noted some drawbacks of previous algorithms and then proposed two new *WSDA*s, namely, Supervised Forward and Backward Ranking (*SFBR*) and Unsupervised Forward and Backward Ranking (*UFBR*). Our experimental results, evaluated on three large datasets, demonstrated the superior performance of *SFBR* in both spam demotion and spam detection. The very promising results obtained by the *USPM* on the WEBSPAM-UK2006 dataset implied that, to a certain extent, it is possible to design unsupervised *WSDA*s that yield comparable results to supervised algorithms in the spam demotion task.

There are still some issues that need further study. For example, the performance of Inverse-PageRank on WEBSPAM-UK2006 suggested that the number of outgoing links may be an indication of spam. However, we cannot confirm this given the limitations of the datasets used. Another issue that has been mentioned in the paper is the score propagation for an unconnected graph. In such a case, the most commonly used seed selection methods are not suitable because of their inherent bias to large connected components. In a future study, we will investigate this problem by exploring local features of connected components to adjust the score propagation approach within them.

Acknowledgements This research is support by Academic and Technological Leadership Training Foundation of Sichuan province, P.R. China.

References

Al-Kabi, M., Wahsheh, H., Alsmadi, I., Al-Shawakfa, E., Wahbeh, A., & Al-Hmoud, A. (2012). Content-based analysis to detect Arabic web spam. *Journal of Information Science, 38*(3), 284–296.

András, A., Benczúr, C. C., Erdélyi, M., Gyöngyi, Z., Masanes, J., & Matthews, M. (2010). ECML/PKDD 2010 discovery challenge data set. Crawled by the European Archive Foundation.

Baeza-Yates, R., Boldi, P., & Castillo, C. (2006). Generalizing pagerank: Damping functions for link-based ranking algorithms. In *Proceedings of the 29th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 308–315).

Becchetti, L., Castillo, C., Donato, D., Leonardi, S., & Baeza-Yates, R. A. (2006). Link-based characterization and detection of web spam. In *AIRWeb* (pp. 1–8).

Becchetti, L., Castillo, C., Donato, D., Leonardi, S., & Baeza-Yates, R. (2008). Web spam detection: Link-based and content-based techniques. In *The European integrated project dynamically evolving, large scale information systems (DELIS): Proceedings of the final workshop* (Vol. 222, pp. 99–113).

Castillo, C., Chellapilla K., & Denoyer L. (2008). Web spam challenge 2008. In *AIRWeb*, 2008.

Castillo, C., Donato, D., Becchetti, L., Boldi, P., Leonardi, S., Santini, M., et al. (2006). A reference collection for web spam. *ACM Sigir Forum, 40*(2), 11–24.

Caverlee, J., & Liu, L. (2007). Countering web spam with credibility-based link analysis. In *Proceedings of the twenty-sixth annual ACM symposium on principles of distributed computing* (pp. 157–166).

Chandra, A., Suaib, M., & Beg, D. (2015). Web spam classification using supervised artificial neural network algorithms. arXiv preprint arXiv:1502.03581.

Chellapilla, K., & Chickering, D. M. (2006). Improving cloaking detection using search query popularity and monetizability. In *AIRWeb* (pp. 17–23).

Convey, E. (1996). Porn sneaks way back on web. The Boston Herald, 028.

Cormack, G. (2007). Content-based web spam detection. In *Proceedings of the 3rd international workshop on adversarial information retrieval on the web (AIRWeb)*.

Cormack, G. V., Smucker, M. D., & Clarke, C. L. (2011). Efficient and effective spam filtering and re-ranking for large web datasets. *Information Retrieval, 14*(5), 441–465.

Diligenti, M., Gori, M., & Maggini, M. (2004). A unified probabilistic framework for web page scoring systems. *IEEE Transactions on Knowledge and Data Engineering, 16*(1), 4–16.

Gyongyi, Z., & Garcia-Molina, H. (2005). Web spam taxonomy. In *First international workshop on adversarial information retrieval on the web*.

Gyöngyi, Z., Garcia-Molina, H., & Pedersen, J. (2004). Combating web spam with trustrank. In *Proceedings of the thirtieth international conference on very large data bases* (Vol. 30, pp. 576–587).

Henzinger, M. R., Motwani, R., & Silverstein, C. (2002). Challenges in web search engines. *ACM SIGIR Forum, 36*(2), 11–22.

Hochstotter, N., & Koch, M. (2009). Standard parameters for searching behaviour in search engines and their empirical evaluation. *Journal of Information Science, 35*(1), 45–65.

Krishnan, V., & Raj, R. (2006). Web spam detection with anti-trust rank. In *AIRWeb* (Vol. 6, pp. 37–40).

Liu, X., Wang, Y., Zhu, S., & Lin, H. (2013). Combating Web spam through trust–distrust propagation with confidence. *Pattern Recognition Letters, 34*(13), 1462–1469.

Marchiori, M. (1997). The quest for correct information on the web: Hyper search engines. *Computer Networks and ISDN Systems, 29*(8), 1225–1235.

Ntoulas, A., Najork, M., Manasse, M., & Fetterly, D. (2006). Detecting spam web pages through content analysis. In *Proceedings of the 15th international conference on World Wide Web* (pp. 83–92).

Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). The PageRank citation ranking: Bringing order to the web.

Radlinski, F. (2007). Addressing malicious noise in clickthrough data. In *Learning to rank for information retrieval workshop at SIGIR*.

Silverstein, C., Marais, H., Henzinger, M., & Moricz, M. (1999). Analysis of a very large web search engine query log. *ACM SIGIR Forum, 33*(1), 6–12.

Spirin, N., & Han, J. (2012). Survey on web spam detection: Principles and algorithms. *ACM SIGKDD Explorations Newsletter, 13*(2), 50–64.

Wu, B., Goel, V., & Davison, B. D. (2006). Propagating trust and distrust to demote web spam. MTW, 190.

Yahoo! Research. (2007). Web spam collections. http://barcelona.research.yahoo.net/webspam/datasets/ Crawled by the Laboratory of Web Algorithmics, University of Milan, http://law.dsi.unimi.it/.

Zhang, X., Wang, Y., Mou, N., & Liang, W. (2011). Propagating both trust and distrust with target differentiation for combating web spam. In *AAAI* (pp. 1292–1297).