SpotRank: A Robust Voting System for Social News Websites

Thomas Largillier
Univ Paris-Sud, LRI;
CNRS; INRIA;
Orsay, F-91405.
thomas.largillier@lri.fr

Guillaume Peyronnet
Nalrem Medias;
Boulogne-Billancourt,
F-92100.
guillaume@nalrem-medias.com

Sylvain Peyronnet
Univ Paris-Sud, LRI;
CNRS; INRIA;
Orsay, F-91405.
syp@lri.fr

ABSTRACT

In a social news website people share content they found on the web, called news, then vote for those they like the most. Voting for a news is then considered as a recommendation, and news with a sufficient number of recommendations are displayed on a front page. Malicious users of such websites boost their own content by manipulating the votes. We present SpotRank, an algorithm that can demote the effect of manipulations, thus leading to a better quality of service. We also present a website that implement this algorithm and show evidence of the efficiency of the approach, both from a statistical and human point of view.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Spam demotion

General Terms

Algorithms, Design, Experimentation, Human Factors

1. INTRODUCTION

In the last years, the way people interact with each others on the Web has drastically changed. Websites now provide information which is an aggregation of user-generated content, generally filtered using social recommendation methods to suggest relevant documents to users. The most known example of such a website is *Digg*[1]. This is a social news website: people share content they found on the web through the Digg interface, then users can vote for the news they like the most. Voting for a news is then considered as a recommendation, and (according to the result of a non disclosed algorithm) news with a sufficient number of recommendations are displayed on Digg's front page.

[1] `http://digg.com`

WICOW'10, April 27, 2010, Raleigh, North Carolina, USA.

Digg has been launched in November 2004, and since then numerous Digg clones (generally denoted as *Digg-like*) were created by webmasters. This huge success can be explained by the amount of traffic such a website aggregate and redistribute. Indeed, being on the front page of a website such as Digg seems to be very interesting since repeated testimonies amongst webmasters state that thousands of unique visitors are obtained within one day for a website on the front page of Digg (or similar sites). Since most websites follow an economic model based on advertisement, obtaining unique visitors is the best way to improve the income. It is then tempting for a user to use malicious techniques in order to obtain a good visibility for his websites.

A malicious technique is explained in details by Lerman in [9] where she recall the Digg 2006 controversy. This controversy arises when a user posted on Digg an analysis proving that the top 30 users of Digg were responsible for a disproportionate fraction of the front page (latter studies ensures that 56% of the front page belongs to the top 100 users only). This means that the top users are acting together in order to have their stories (e.g websites they support) displayed on the front page. The controversy led to a modification of the Digg algorithm in order to lower the power of this so-called *bloc voting* (collusion between a subset of users).

Since 2006, malicious users became more and more efficient (see for instance the paper of Heymann *et al.* [4]). Cabals (collusion of large group of users that vote for each others) have been automatized using daily mailing list, some users post hundreds of links in order to flood the system, others have several accounts and thus can vote for themselves (using several IP addresses) etc. To the best of our knowledge, no social news website implements a robust voting scheme that avoid the problem of dealing with malicious users while still providing a high quality of service (i.e. providing relevant news to users).

The main contributions of this paper are:

The design of the SpotRank algorithm. SpotRank is a set of heuristic techniques whose aim is not to detect and suppress malicious voting behaviors in social news website, but rather demote the effect of these behaviors, thus leading to lower the interest of such manipulations for spammers. SpotRank is built over *ad-hoc* statistical filters, a collusion detection mechanism and also over the computation of the *pertinence* of voters and proposed news.

A strong experimental analysis. We present a website that implement this algorithm and show evidence of the efficiency of the approach, both from a statistical and human

point of view. This analysis is twofold : we give evidence that using SpotRank we maintain a behavior for the social news website which correspond to a regular behavior, and we also provide a study of the perceived quality of the algorithm on 114 users of our experimental platform (*via* a comparison with others french social news websites).

The structure of the paper is as follows. In section 2 we give some insight about the very few related work. Then, in section 3, we present our modeling and describe in detail the SpotRank algorithm. Last, we give in section 4 a complete experimental analysis of our method.

2. RELATED WORK

Regarding the analysis of social news websites there are only a few research papers available. The work done by Kristina Lerman (and her coauthors) in many papers [9, 11, 10, 14, 8] is probably the most extensive done in this field. These papers analyse the behavior of users and content in social sites such as Digg and *Flickr*[2]. Abstract modeling of users is done and allows to infer the dynamics of users' rank [8], but also to predict which news can obtain good ranking according to the first votes [11]. However, this work is analytic, the goal is to understand how social news websites work. We, on the other hand, aim at designing a robust voting scheme in an adversarial environment, thus our approach is normative.

In their paper [4], Heymann *et al.* present a survey on spam countermeasures for social websites. They sorted three categories of such countermeasures: identification-based methods (i.e. detection of spam and spammers), ranked-based method (i.e. demotion of spam) and limit-based method (preventing spam by making spam content difficult to publish). Clearly SpotRank falls in the scope of ranked-based method since our goal is to reduce the prominence of content that benefit from malicious votes.

Bian *et al.* [1] describe a machine learning based ranking framework for social media that is robust to some common forms of vote spam attacks. Some other work focusing on manipulation-resistant system, and using a notion close to the one of pertinence, can be found in [16].

A related field of research is the detection of click fraud in the Pay Per Click (PPC) advertising market, but also in web search ranking. In PPC, webmasters display clickable advertisements on their website and are paid for each click going through the ad. In web search ranking, the more a link to a website is used, the higher the site is ranked. For instance Jansen [6] give details of the impact of malicious clicks in PPC while Metwally *et al.* [12], Immorlica *et al.* [5] give strong analysis of the phenomenon together with algorithms to cope with it. Radlinski and Joachims [15] focus on randomized robust techniques that infer preferences from click-through logs.

The problem of giving to the users of a community a good selection of news seems to be a recommendation problem. Cosley *et al.* [2] study the relation between recommendation systems and users, while Lam and Riedl [7], and O'Mahony *et al.* [13] address the problem of malicious users and robustness of systems. However, recommendations by friends has been proven to always be better than recommendations using automatic systems (see for instance the papers of Sinha and Swearingen [17]). To overcome this problem, researchers from the recommendation systems field introduce the notion of trust as a reflection of the users similarity.

In this paper, we focus on completely different techniques that demote votes that are malicious, or done by users known to be malicious. Our approach does not use machine learning methods and is based on the notion of *pertinence*. It is worth noting that despite the lack of research papers in this field, there are probably a lot of undisclosed work going on in the social news websites' teams.

[2] `http://www.flickr.com/`

3. SPOTRANK ALGORITHM

In this section we first present the framework on which Spotrank is built together with its principle. We then describe independently each step of the algorithm.

3.1 Framework and principle

In this paper, we consider that the voting system (SpotRank) is used by a community of users belonging to the set $\mathcal{U}$. Any user of the community can propose its own news (or content), that we will call *spots*. The set of spots is denoted by $\mathcal{S}$. Any user of the community can vote for a spot. A vote is a triple (u, s, v) where $u, v \in \mathcal{U}$ and $s \in \mathcal{S}$. The set of all votes is noted $\mathcal{V}$. For the sake of clarity we introduce some notations:

$\mathcal{V}_u = \{(w, s, v) \in \mathcal{V} \mid w = u\}$

$\mathcal{V}_{uu'} = \{(w, s, v) \in \mathcal{V} \mid w = u, v = u'\}$

$\mathcal{V}_s = \{(u, t, v) \in \mathcal{V} \mid t = s\}$

$\mathcal{V}_u$ denotes votes made by u, $\mathcal{V}_{uu'}$ the votes made by u for a spot proposed by u', and $\mathcal{V}_s$ the vote made by all users in favor of the spot s. This modeling is rather theoretical, in practice we have access to a lot more information on users, votes and spots. In the following, we will access this information through functions that will be either clearly defined by the context or explicitly just before their use.

We can now schematize the SpotRank method. It is important to know that it is based on two key notions. The first one is that two votes do not necessarily have the same value. Indeed, we will assign, depending on many factors, a score to each vote. This will induce a score for each spot (the sum of the score of each vote in favor of this spot). The higher the score of a spot, the closer to the first place (e.g. the top of the front page) is the spot. A large part of SpotRank is the score computation mechanism. The other key notion is the *pertinence*. The pertinence of a user depends on the pertinence of the spots he voted for, and *vice versa*. A part of the score update of a spot will depend on the pertinence of the user that votes for this spot. Last, an additional mechanism is used in order to avoid large scale manipulations: a method to detect cabals (i.e. group of users that repeatedly vote one for each other).

Finally, figure 1 depicts the voting process of SpotRank for a given spot. The method is working as follows:

A user proposes a spot. The score of this spot is initialised according to several criteria (all related to the known behavior of the user).

Users vote for the spot. Each vote induces an update on the spot's score and pertinence, but also of the pertinence of users that previously voted for this spot. The score of the spot is then used by a social news website in order to rank published content.

Periodically, an algorithm that detects collusion between users outputs clusters of potential malicious users. These clusters are used for computing the score of a vote.

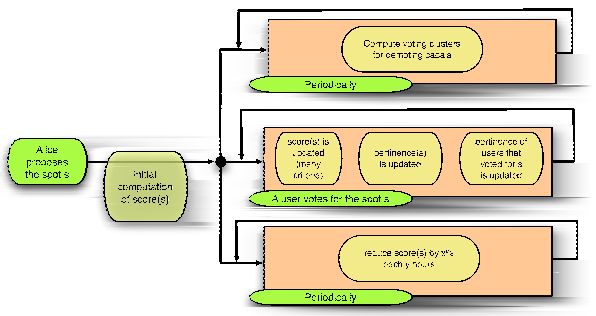

Figure 1: Principle of our method

To avoid old very strong spots to stay forever at the top of the ranking, we reduce the score of a spot **periodically**.

In the following we describe in details each of these steps. Each time a practical parameter is used, we denote it by α_i (where $i \in \mathbb{N}$) and discuss on its real actual value in practical applications. Actual values are set according to a back and forth method: we set a value, test the robustness of the method with this value, modify it if necessary, until we obtain a good quality of result.

3.2 Proposing a spot

The first step of our method is the proposition of a spot. The main threat at this point for social news website is the flood of the system by some malicious users.

When a user proposes a spot it is necessary to initialize its score. In an ideal world any value can be used for this initialization, but we will use this step to demote spots proposed by frenetic spot posters. The initial score will then depend on two factors: first, the frequency at which the user proposes spots and second, the frequency at which new spots come from the user's IP address.

More precisely, the initial score of a spot s is calculated with the following formula:

$$\text{init\_score}(s) = f(n) * c_{IP}(m),$$

where n is the number of spots proposed by the user in the last 24 hours, m is the number of spots previously posted from the user's IP in the last 20 minutes. f and c_{IP} are defined as follows:

$$f(n) = \begin{cases} 100 & \text{if } n < \alpha_0 \\ 50 & \text{if } \alpha_0 \leq n < 2\alpha_0 \\ 10 & \text{if } 2\alpha_0 \leq n < 4\alpha_0 \\ 0 & \text{otherwise} \end{cases}$$

$$\text{and } c_{IP}(m) = \max\left(0, 1 - \frac{m}{\alpha_1}\right).$$

In practical applications α_0 and α_1 will be small integers that depend on the number of visitors of the social news website. We recommend to use $\alpha_0 = 2$ and $\alpha_1 = 10$.

With this formula for initializing the score of a new spot, we prevent the effective "spot bombing" from spammers. Indeed the proposed spot initial score will drop fast until it has value zero. The $c_{IP}(m)$ coefficient also helps to track down spammers that use several accounts to pollute the spot pool. If we add to a social news website a mandatory identification in order to propose a spot, it becomes meaningless to use several accounts (IP spoofing is then useless).

3.3 Voting for a spot

Once a spot has been proposed to the community it can be "pushed" to the front page (i.e. put in the top ranked news). The spots ranking is done according to their scores. The voting part is the one requiring the most attention since it is where the spammers will concentrate all their attacks. We propose a set of filters whose aim is to counter all the attacks a spammer could think of. The idea here is simple: a vote has a base value which is the pertinence of the voter. This value is then modified according to several criteria to provide the actual value of a vote (its score). The score of the vote is then added to the current score of the spot in order to obtain its new score.

Base value of a vote: *pertinence*. The key notion of our voting system is the *pertinence* of users and spots. We denote the pertinence of user u_i by $pert(u_i)$. Similarly, the pertinence of a spot s_j is $pert(s_j)$.

DEFINITION 1. *The pertinence of a user without voting history (i.e. a new user) is a constant α_3.*

The pertinence of a user u_i with a voting history is:

$$pert(u_i) = \frac{1}{|\mathcal{V}_u|} \sum_{(u,s,v)\in\mathcal{V}_u} pert(s),$$

where

$$pert(s) = \frac{\text{score}(s)}{|\mathcal{V}_s|}.$$

The pertinence of a user u is thus the mean value of the pertinence of the spots u voted for. The pertinence of a spot is its score divided by the number of votes it received (i.e. this is the mean value of the votes for s). In our experiments we set $\alpha_3 = 100$ to match the value of a fresh legitimate spot.

We can now define the base value of a vote as the pertinence of the user that votes. This value is weighted by several coefficients that we can now describe, each of this coefficient can be seen as a response to a specific type of attacks.

High frequency voting. Most of the time, spammers try to promote a lot of low quality news. All the gain of their manipulations is more due to mass effect than to the promotion of only one peculiar web page. Thus they have no other way of voting than using burst voting. It means that a typical spammer votes for a lot of spots in a short amount of time. To demote this effect we will weight the score of a spot by a coefficient $\text{freq}(u)$ that depends on the user's voting frequency.

DEFINITION 2. *Let $n = |\mathcal{V}_u|$, $\text{freq}(u)$ is defined as*

$$\text{freq}(u) = \begin{cases} 1 & \text{if } n < 2 \\ \min(1, \frac{\text{date}(v_n) - \text{date}(v_1)}{\alpha_4 * n}) & \text{otherwise} \end{cases}$$

α_4 is the time interval that is reasonable between two votes.

Abusive one-way voting. In order to decoy manipulation detection, a typical spammer uses several accounts: one clean account to propose spots, and several disposable accounts to vote for the spots proposed by the clean account. To reduce the score of votes made in this spirit, we define a coefficient $fp(u, u')$ whose goal is to take into account the particular frequency of systematic one-way voting from a user u to a user u'.

DEFINITION 3. *Let $u, u' \in \mathcal{U}$, fp is defined as*

$$fp(u,u') = 1 - \frac{|\mathcal{V}_{uu'}|}{|\mathcal{V}_u|}$$

With this coefficient, users that vote only for one specific user will have their vote becoming useless.

Quick voting. Spamming is a large scale activity so it is unthinkable for a spammer to stay a long time on one given website. The behavior is then to propose a spot and to quickly vote for it using several accounts. This is not a natural behavior since the time interval between the proposition and the vote is too short for a human to even look at the website associated to the spot.

To avoid quick voting we block any vote in the first minute of appearance of the spot s on the site and after that we use a stair function based on $\text{date}(s)$ and t the current time. This function (called time) is defined as follows.

DEFINITION 4.

$$\text{time}(s) = \begin{cases} 0.3 & \text{if } t - \text{date}(s) < 120 \\ 0.5 & \text{if } t - \text{date}(s) < 240 \\ 0.7 & \text{if } t - \text{date}(s) < 420 \\ 0.9 & \text{if } t - \text{date}(s) < 540 \\ 1 & \text{otherwise} \end{cases}$$

Multiple avatars and physical community. As said previously, a typical spammer will have many accounts, sometimes he will also have automatic voting mechanisms. These voting bots are often located on only a few servers, so they share the same IP address (or only very few IPs addresses). It is then interesting to have a coefficient that demotes votes for a given spot if they come from the same IP address. One could object that legitimate users can share the same IP address. Our opinion on this matter is that when legitimate users belonging to the same IP address vote for the same spot it is a kind of manipulation (one can think of students of the same university that vote for one of them).

Therefore, for a same spot, votes coming from the same IP address receive a decreasing coefficient coeff_{IP} depending on the number n of previous votes from this IP address:

DEFINITION 5.

$$\text{coeff}_{IP} = (\alpha_5)^n$$

In practical applications α_5 is a real number between 0 and 1 (typically $\frac{2}{3}$ in our case).

Avoiding the voting list effect. The main threat for social news website is the existence of cabals. A cabal is a group of people that unite their efforts in order to promote their own spots. There exists highly organised cabals whose goal is to manipulate ranking of social news websites. This is classically done through daily mailing lists. The daily mail contains a list of news for which votes are required. Users of such lists can propose their own news to the list depending, most of the time, of the number of votes they made for other members of the cabal.

In the next subsection we present our method for detecting cabals (that we call clusters). But here we assume that we already detect these clusters. Such a cluster is a list of users that periodically vote one for an other.

To slow down the effect of the cabals we proceed as follows: if a user u votes for a user u' and both users are in the same "cluster" then the value of the vote is weighted by the inverse of the size of this "cluster". This lead to the definition of the following coefficient.

DEFINITION 6. *let* $\text{cluster}(u)$ *be the cluster to which user u belongs (if any), then:*

$$\text{clust}(u,u') = \begin{cases} 1 & \text{if } \text{cluster}(u) \neq \text{cluster}(u') \\ \frac{1}{|\text{cluster}(u)|} & \text{otherwise} \end{cases}$$

Summary: computation of the actual score of a vote. We can now define the score of a vote. This is actually the base value score weighted by all the coefficients we defined above.

DEFINITION 7. *the score of a vote v from the user u for the spot s posted by the user u' is:*

$$\text{score}(v) = pert(u) \cdot \text{freq}(u) \cdot fp(u,u') \cdot \text{clust}(u,u') \cdot \text{time}(s) \cdot \text{coeff}_{IP}$$

It now remains to define the score of a spot according to this definition.

Computation of the score of a spot. The score of a spot is simply the sum of all votes for this spot and of the initial score of the spot. This quantity is however weighted by a multiplicative coefficient that depends on the age of the spot. This time decay is used to promote new spots against old strong spots (it is not interesting that popular news stay forever on top of the ranking).

DEFINITION 8. *The score of the spot $s \in \mathcal{S}$ is defined as:*

$$\text{score}(s) = \text{time\_decay}(s) \cdot \big(\text{init\_score}(s) + \sum_{v \in \mathcal{V}_s} \text{score}(v)\big)$$

The score of a spot s is updated each time a user votes for it, but also periodically since the value of time_decay varies over time. In practical applications, we define the time decay as:

DEFINITION 9. *Let d be the age (in days) of the spot.*

$$\text{time\_decay}(s) = \begin{cases} 1 & \text{if } d \leq 2 \\ 0.8^d & \text{if } d > 2 \end{cases}$$

The decay start after 2 days to give fresh spots an advantage over old ones. The value 0.8 comes from an experiment on the better value to ensure a sufficient turn-over on the front-page of the social news website.

On a classical social news website using SpotRank, all spots are ranked according to their score (the higher the better).

3.4 Detecting cabals

In this subsection, we present our algorithm for the detection of collusion of voters. It is a fair idea to use the weighted directed graph of votes (nodes represent voters, arcs correspond to votes and weights to the number of votes between two users). The standard approach to identify group of users in this graph is then to cluster it. State-of-the-art techniques such as the ones of [18, 3] are not efficient on large graphs. The graph underlying a social news website can quickly attain a huge size, so we cannot use these clustering techniques.

Instead, we propose here to regroup people that massively vote between themselves. Therefore we use the following algorithm that should be run regularly to identify new cabals and actualize the existing ones.

Collusion detection for user u
1. Let $E = \{(v,k)/(\mathcal{V}_{u,v} \neq \emptyset \wedge k = |\mathcal{V}_{u,v}|\}$
   Sort E according to k in decreasing order.
2. Put the first p users (according to this sorting) into a set $Fav_p(u)$.
3. Sort alphabetically the set $Fav_p(u)$ with u's ID included.
4. $\forall v \in Fav_p(u)$, if $|Fav_p(u) \cap Fav_p(v)| > \alpha_6$ then u and v are in the same group.

This algorithm should be run for each user in order to assign this particular user to a cluster (potentially a cluster of size 1). Let $n = |\mathcal{U}|$, the complexity of the algorithm is $\mathcal{O}(n\log(n))$ for the step 1, $\mathcal{O}(1)$ for the step 2, $\mathcal{O}((p+1)\log(p+1))$ for the step 3, and $\mathcal{O}(p^2)$ for the step 4. The total complexity is then $\mathcal{O}(n\log(n)+p^2)$ which is, in practice, $\mathcal{O}(n\log(n))$ since p is a fixed parameter. In our experiment, $p = 5$ and $\alpha_6 = 3$. Those parameters have rather small values, it can be explained by the fact that our community is still small (200 users).

After running this algorithm we store the groups along with their size to reuse it during the vote phase.

4. EXPERIMENTS

We present two different evaluations of SpotRank. The first is a statistical analysis of the output of our method: distribution of users, votes, pertinence of both users and spots, ranking, cluster sizes, etc. The second evaluation is a investigation of the human perceived quality of our ranking method.

In order to collect data about the behavior of SpotRank we created a social news website (`http://www.spotrank.fr`). Spotrank.fr strictly implements the method presented in this paper and ranks the spots according to their score (computed as presented in section 3). The website has been launched on July the 9th. The data we use for the paper were collected the 10/26/2009. Since its launch, the website received around 15600 visits, had served around 43000 page views. The bounce rate is 67.85% and the average time spent by a visitor on the website is 2:37 minutes. The presence of spammers is effective, we estimate (by hand) that at least 10 to 15% of our 200 registered accounts belong to spammers. Log files are available for the reviewers upon request.

4.1 Log analysis of spotrank.fr

Our log files contain all the information about spots and users and allow us to show accurately the behavior of our method in a real adversarial environment. We present our analysis through several figures. The information we used for this experimentation was collected between the 23rd of July and the 26th of October.

Figures 2, 3 and 4. These figures show similar data at three different moments of our analysis: at the beginning (07/23), the 09/08 and at the end (10/26). They all represent the percentage of users of SpotRank that have a given pertinence. For instance the first bar on the left of figure 2 means that around 7% of the users have a pertinence between 0 and 5. We can see that as time goes and the number of users grows the pertinence of the users tends to spread more. The percentage of users at the right of all graphics represents the new users that never voted for anything so they keep their initial maximum pertinence. Even if distribution of users regarding the pertinence seems almost uniform, we can see that most users have a pertinence between 15 and 50. Thus we define two categories of users, the non-relevant users whose pertinence is less than 10 and the relevant users whose pertinence is greater than 50 but don't include the newcomers in order to make sure that this good pertinence is not a consequence of a clean start. Being in the non-relevant category does not mean for a user that he votes for spots others don't like. It means that the user votes often for spots with low pertinence. It is likely that this category contains mainly spammers.

Figure 5. The two curves on this figure show the evolution of the proportion of users belonging to both the non-relevant and the relevant categories on a period of 3 months. We can see that the percentage of non-relevant users including spammers is decreasing while the percentage of relevant users is increasing. This could be explained by spammers being discouraged but more likely by the arrival of new relevant users. The increase of the size of the relevant category can also be explained by the fact that at the launch of the website, spammers propose spots faster than legitimate users. This means that at first most of the spots have a low pertinence, inducing a low pertinence for voters.

Figure 6. This histogram presents the number of users that proposed a given number of spots during the 3 months of the experimentation. It can be seen that the majority of users proposes a few spots (less than 3). There are few people with a oddly high number of proposed spots. We checked by hand the top proposers. Amongst them, the first three (with respectively 591, 440 and 108 spots) are clearly spammers while the fourth (99 spots) is a borderline user that proposes every single post of his blog.

Figure 7. The figure depicts the number of users w.r.t. the number of votes. Pay attention to the fact that the Y-scale is logarithmic. We can see that most users don't vote a lot: more than 80% of them had voted less than 10 times. The people that vote the most are clearly the ones we suspect to be spammers. If we look for instance at the outlier with 533 votes, this is without surprise the spammer we spotted on the previous figure. It is a user that proposes a lot of spots and vote only for his own spots. It does not appear in the figures, but they are users that does not propose spots (or only very few) and that vote a lot.

Figure 8. On a social news website such as spotrank.fr there are only a few slots available for promoting highly popular spots. Is this a problem? Is the pressure on spots too high to ensure a fair access to the front page for spots that deserve it? Figure 8 gives the answer by showing the behavior of spots' score during their first 48h of existence (i.e. before they undergo the exemption owed to the time decay). The max (resp. min) curve give the score of the most (resp. least) popular spot at the time the measure is done. The average curve give the mean value of scores during the first 48h. We can clearly see that only a few spots become popular. We can also infer from this figure is the average value of votes for the most popular spot. Indeed, most popular spots received around 15 votes, meaning that the average score of votes for these popular spots is around 25 (to compare to 100, the maximum possible score of a vote, but also to 10, the threshold beyond which users are not considered relevant anymore).

Figure 9. This last figure shows the number of votes that share a given score. It is clear that most of the votes ($\approx$1600) have very low score. It is not a surprise since we already have exhibited a spammer with 533 votes that vote only for

himself and with high frequency, meaning that several of our filters act to preclude the score to be high. Most legitimate users seems to have votes with score between 5 and 50, and only a few have very high score.

To summarize, the figures that we provide in this subsection show clear evidence of the effectiveness of the method: spammers are detected, and the score of votes seems to be adapted to avoid manipulations.

4.2 Human Evaluation

Even with a very strong analysis of the log files, it is impossible to judge the quality of the filtering of our method. Indeed, the algorithm consists in filtering news w.r.t. the way people vote, it is not content related. To cope with this issue we decided to gather some feedback from the website visitors. Since an absolute judgement is impossible to obtain without a long debate on what is the quality of a website, we choose to compare the top "stories" of three social news website. The first is of course spotrank.fr which implement the method presented in this paper, the other two are two of the three major competitors in the field in France. The interest of the two chosen social news website is that they use automatic method to filter news, and also that the human moderation has mainly the goal to suppress non legal content. The way these two competitors are filtering news is mostly unknown since those are business websites. It is just known that one of them is giving more weight to news for which there are a lot of votes in a short time interval. From now on, they will be denoted as comp1 and comp2. Our survey protocol is the following. To have relevant results we periodically collected the first five spot on spotrank.fr together with the top 5 of the two major french social news websites comp1 and comp2. We then automatically generate disposable web pages containing a shuffle of this list of 15 news. Each web page is then sent to a volunteer who has to tell for each news if,

Yes, it is relevant for the news to appear on the front page of a social news website.
No, it is not relevant for the news to appear on the front page of a social news website.
DnK, he is not able to determine if the news deserve to be on the front page or not.
Err, The news was not accessible when he tried.

We collected the first five news of each website during a period of 3 months, and 114 persons participated to the poll. We now present our experimental results.

Figure 10. This figure could be considered as a summary of the results of the poll. For each competitor it presents the number of **Yes**, **No**, **Dnk** and **ERR**. The number of **ERR** that appears in surveyed people answers is not of interest since this is an external factor that applies for all three social news website. However a higher rate of error could indicates links to unreliable site. Pay attention to the fact that for each competitor, each surveyed person is giving 15 answers, so the total number of answers is 1710.

The important point is that SpotRank outperforms both competitors whatever the criterion. Our method received 344 **Yes** while comp1 and comp2 received respectively 270 and 177 **Yes**. The performances of comp1 (resp. comp2) are only 78.5% (resp. 50%) of those of SpotRank, thus surveyed people think that the ranking given by SpotRank is of higher quality than the two others. Concerning the **No** answer the situation is similar: this time the lower is the better since this means that the top spots are considered not legitimate and SpotRank received 174 **No**, while comp1 and comp2 received 237 and 247 such answers. Last, 44 **DnK** were received by SpotRank. This is again a better achievement than comp1 and comp2 and this means that the filtering of SpotRank gives clearer results (only a few borderline spots).
Figure 11. In the previous figure, the behaviors of all users were merged in the counting of each type of answer while here we consider the opinion of each surveyed person. a social news website is considered first if the number of **Yes** he received from a peculiar answerer is greater than those received by the two other competitors. It is second (resp. third) in the case where it received the second (resp. third) number of **Yes**. SpotRank is in first position two times (resp four times) more than comp1 (resp comp2), showing again its higher performances. It is naturally second and third less often than the others.
Figures 12 and 13. These figures are similar to the one we mentioned before for the **No** and **DnK** answers. These two figures confirm what we have presented above.

To summarize, this user satisfaction survey show clearly that the filtering of SpotRank is perceived to be of high quality. It is interesting to note that some early adopted spammers have already given up playing with SpotRank (see previous subsection).

5. CONCLUSION

We presented a robust voting system for social news website whose goal is to demote the effect of manipulation. Through a website that implement this algorithm, we show evidence of the efficiency of the approach, both from a statistical and human point of view. SpotRank clearly outperforms real competitors in a real life web ecosystem, proving the interest of the key notions used to design the method (pertinence, frequency filtering and collusion detection).

6. REFERENCES

[1] J. Bian, Y. Liu, E. Agichtein, and H. Zha. A few bad votes too many?: towards robust ranking in social media. In *Proceedings of the 4th international workshop on Adversarial information retrieval on the web*, pages 53–60, New York, NY, USA, 2008. ACM.

[2] D. Cosley, S. K. Lam, I. Albert, J. A. Konstan, and J. Riedl. Is seeing believing?: how recommender system interfaces affect users' opinions. In *CHI '03: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 585–592, New York, NY, USA, 2003. ACM.

[3] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *PROC.NATL.ACAD.SCI.USA*, 99:7821, 2002.

[4] P. Heymann, G. Koutrika, and H. Garcia-Molina. Fighting spam on social web sites: A survey of approaches and future challenges. *IEEE Internet Computing*, 11(6):36–45, 2007.

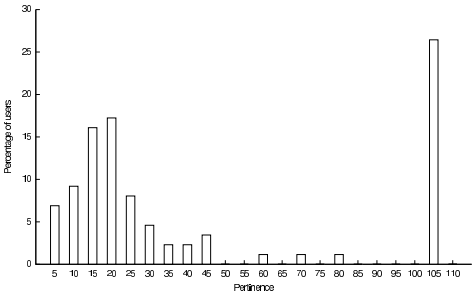

Figure 2: % of users w.r.t. pertinence (1/3)

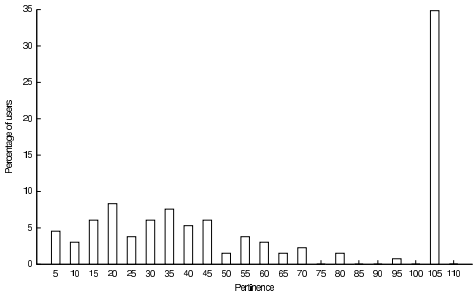

Figure 3: % of users w.r.t. pertinence (2/3)

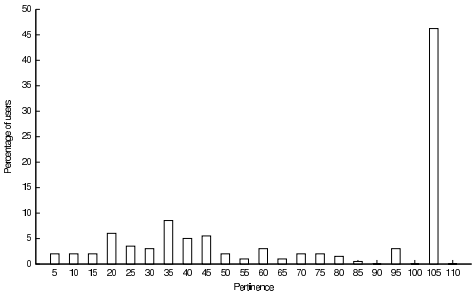

Figure 4: % of users w.r.t. pertinence (3/3)

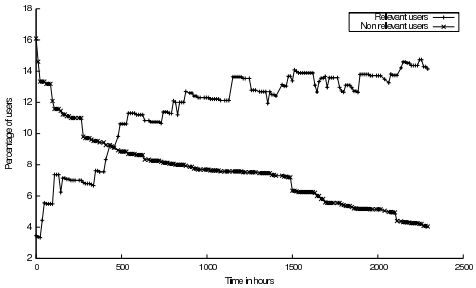

Figure 5: % of low and high pertinent users w.r.t. time

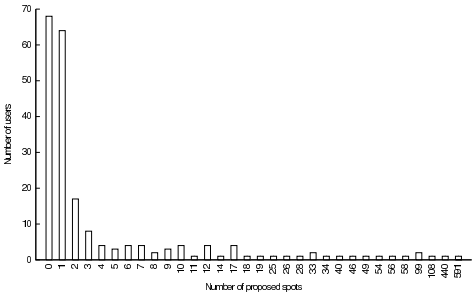

Figure 6: # users versus # proposed spots

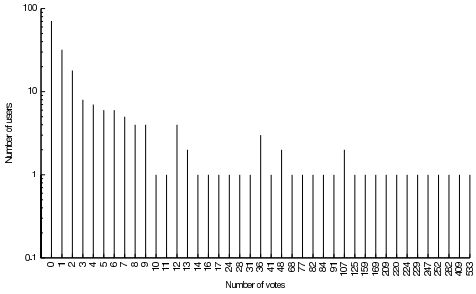

Figure 7: % users w.r.t. # votes

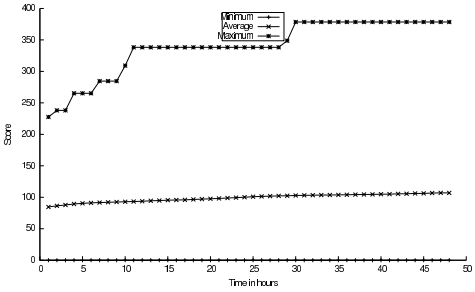

Figure 8: spots score w.r.t. time

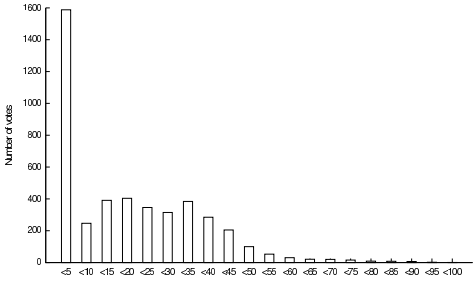

Figure 9: # votes versus their scores

[5] N. Immorlica, K. Jain, M. Mahdian, and K. Talwar. Click fraud resistant methods for learning click-through rates. In *In WINE*, pages 34–45, 2005.

[6] B. J. Jansen. Adversarial information retrieval aspects of sponsored search. In *AIRWeb*, pages 33–36, 2006.

[7] S. K. Lam and J. Riedl. Shilling recommender systems for fun and profit. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 393–402, New York, NY, USA, 2004. ACM.

[8] K. Lerman. Social information processing in news aggregation. *IEEE Internet Computing: special issue on Social Search*, 11(6):16–28, November 2007.

[9] K. Lerman. User participation in social media: Digg study. In *Proceedings of the 2007 IEEE/WIC/ACM International Conferences on Web Intelligence and*

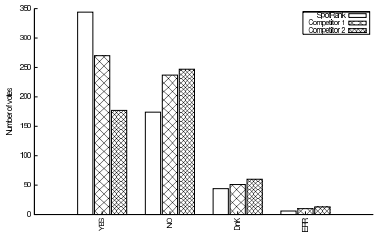

Figure 10: # answers of each type

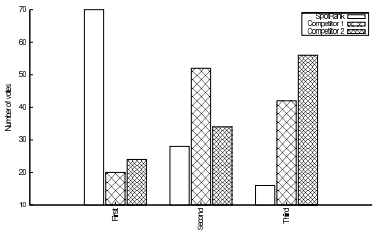

Figure 11: rank w.r.t. the number of Yes

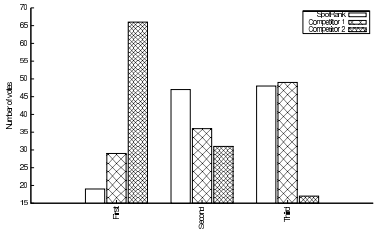

Figure 12: rank w.r.t. the number of No

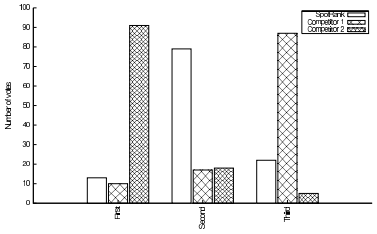

Figure 13: rank w.r.t. the number of DnK

Intelligent Agent Technology, pages 255–258, Washington, DC, USA, 2007. IEEE Computer Society.

[10] K. Lerman. Dynamics of a collaborative rating system. pages 77–96, 2009.

[11] K. Lerman and A. Galstyan. Analysis of social voting patterns on digg. In *WOSP '08: Proceedings of the first workshop on Online social networks*, pages 7–12, New York, NY, USA, 2008. ACM.

[12] A. Metwally, D. Agrawal, A. E. Abbad, and Q. Zheng. On hit inflation techniques and detection in streams of web advertising networks. *Distributed Computing Systems, International Conference on*, 0:52, 2007.

[13] M. O'Mahony, N. Hurley, N. Kushmerick, and G. Silvestre. Collaborative recommendation: A robustness analysis. *ACM Trans. Internet Technol.*, 4(4):344–377, 2004.

[14] A. Plangprasopchok and K. Lerman. Constructing folksonomies from user-specified relations on flickr. In *Proceedings of the International World Wide Web Conference*, April 2009.

[15] F. Radlinski and T. Joachims. Minimally invasive randomization for collecting unbiased preferences from clickthrough logs. In *Conference of the Association for the Advancement of Artificial Intelligence (AAAI)*, pages 1406–1412, 2006.

[16] P. Resnick and R. Sami. The influence limiter: provably manipulation-resistant recommender systems. In *RecSys '07: Proceedings of the 2007 ACM conference on Recommender systems*, pages 25–32, New York, NY, USA, 2007. ACM.

[17] K. Swearingen and R. Sinha. Beyond algorithms: An hci perspective on recommender systems. *ACM SIGIR 2001 Workshop on Recommender Systems*, 2001.

[18] S. van Dongen. *Graph Clustering by Flow Simulation*. PhD thesis, University of Utrecht, May 2000.