Fighting Spam on Social Web Sites

A Survey of Approaches and Future Challenges

In recent years, social Web sites have become important components of the Web. With their success, however, has come a growing influx of spam. If left unchecked, spam threatens to undermine resource sharing, interactivity, and openness. This article surveys three categories of potential countermeasures — those based on detection, demotion, and prevention. Although many of these countermeasures have been proposed before for email and Web spam, the authors find that their applicability to social Web sites differs. How should we evaluate spam countermeasures for social Web sites, and what future challenges might we face?

Paul Heymann,
Georgia Koutrika,
and Hector Garcia-Molina
Stanford University

Social Web sites such as Wikipedia, del.icio.us, and Flickr have gone from being a small niche of the Web to one of its most important components. These sites rely on user-generated content, making them both incredibly dynamic and tempting targets for spam. The notion of spam is subjective, but in this article, we mean either content designed to mislead or content that the site's "legitimate" users don't wish to receive. Spam content can take many forms, possibly manifesting as a document, an annotation, a user profile, or an automated vote. The attacker's motivations largely determine the form, whether advertising, self-promotion, disruption, curiosity, or disparaging a competitor. However, whatever the form, the long-term viability of social Web sites will ultimately be determined in large part by their resistance to spam.

In this article, we survey potential solutions for combating spam on social Web sites, and, by doing so, compare and contrast those with prior solutions to email and Web spam. Our first step is to identify social Web sites' key characteristics as they relate to spam.

Social Web Sites

Social Web sites capture and display content generated by untrusted users, which can range from photos to text to URLs. In

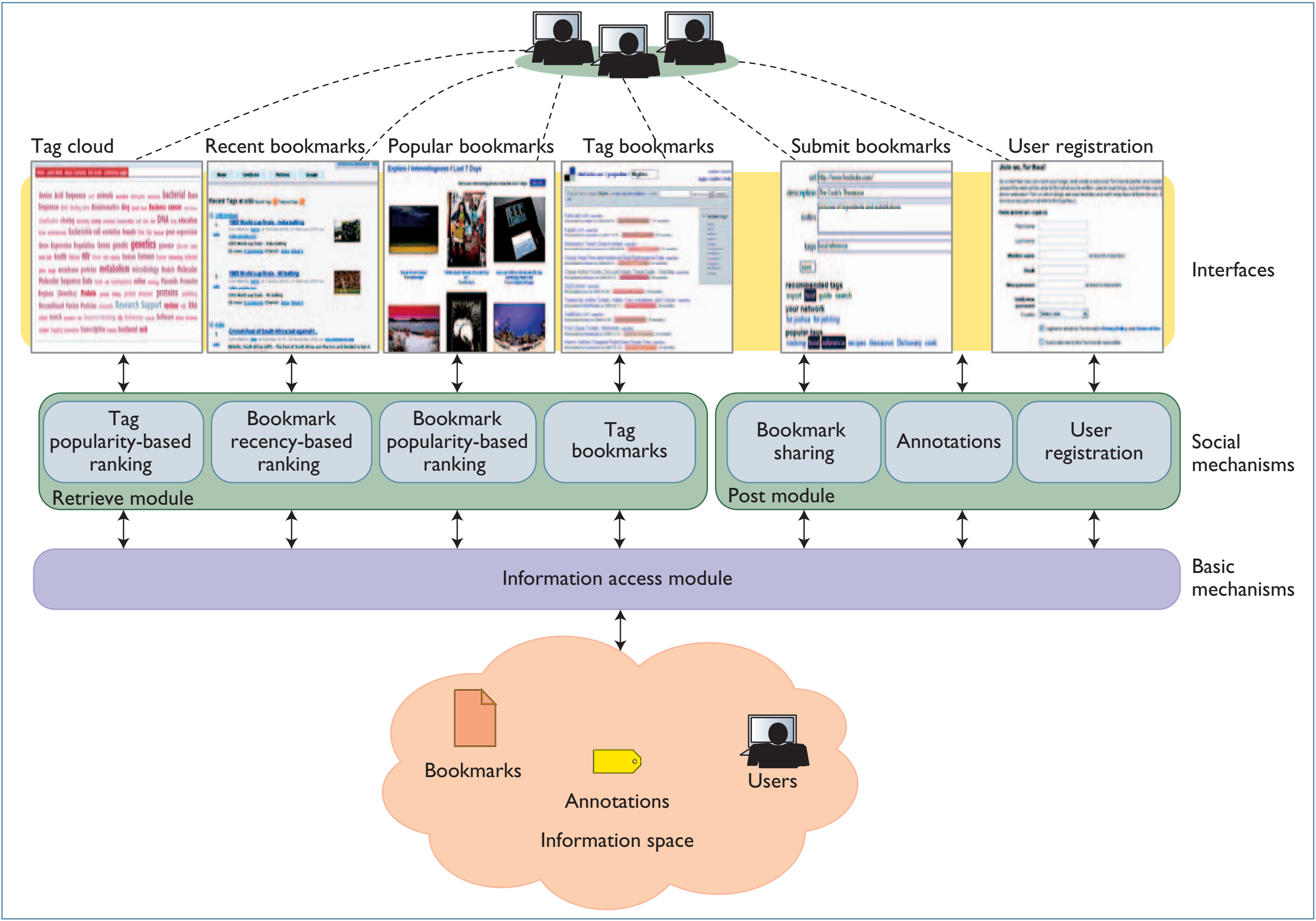


Figure 1. A complete social bookmarking system. Users (top of figure) interact with interfaces that correspond to social mechanisms or well-defined actions on the site. These actions lead to data-level changes in the site's back end.

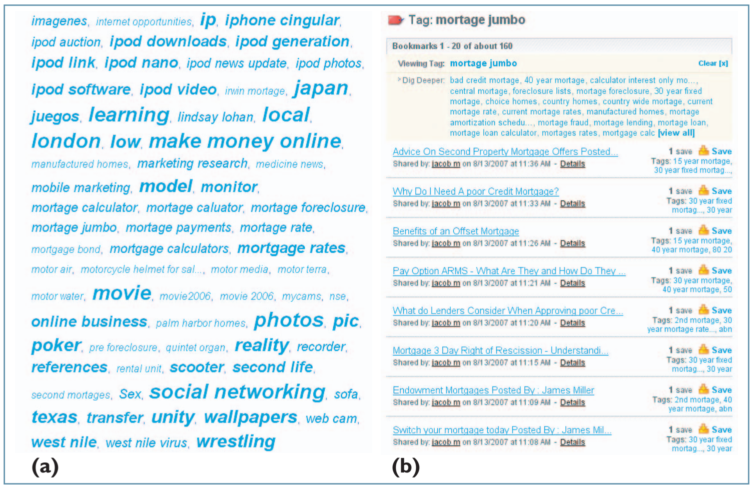
addition, content might include annotations, such as single words (tags), comments, or votes.

Social Web sites have four important characteristics relevant to our discussion of spam:

- *One controlling entity.* A single entity or owner manages the system's content and maintenance.
- *Well-defined interactions.* The controlling entity determines a constrained set of actions available to users. For example, a social bookmarking system lets users contribute bookmarks annotated with tags, and possibly share bookmarks with others, but few other actions might be allowed.
- *Identity.* Content and actions in the system are tied to user identifiers; thus, their source can identify users.
- *Multiple interfaces.* Users have multiple views of site content. In a social bookmarking system, for example, users view the most frequently occurring tags (a "tag cloud"), or the most recently contributed bookmarks. Each view might have a different algorithm determining what content is displayed and in what manner.

These characteristics substantially change the dynamic of the adversarial relationship between service providers and spammers. The controlling entity can enforce spam-fighting techniques that weren't popular enough to gain traction in distributed email and Web systems. It can also make its decisions based on a history of well-defined actions by relatively well-identified users. However, having multiple interfaces means that this entity must protect many fronts — rather than a single one — from spam. This new dynamic means that spammers have many more avenues for attack, whereas service providers have more potential defensive strategies.

To illustrate these characteristics, we describe a simple social bookmarking site. This description will also drive our discussion of spam-fighting techniques.



imagenes, internet opportunities, ip, iphone cingular, ipod auction, ipod downloads, ipod generation, ipod link, ipod nano, ipod news update, ipod photos, ipod software, ipod video, irwin mortage, japan, juegos, learning, lindsay lohan, local, london, low, make money online, manufactured homes, marketing research, medicine news, mobile marketing, model, monitor, mortage calculator, mortage caluator, mortage foreclosure, mortage jumbo, mortage payments, mortage rate, mortgage bond, mortgage calculators, mortgage rates, motor air, motorcycle helmet for sal..., motor media, motor terra, motor water, movie, movie2006, movie 2006, mycams, nse, online business, palm harbor homes, photos, pic, poker, pre foreclosure, quintet organ, reality, recorder, references, rental unit, scooter, second life, second mortages, Sex, social networking, sofa, texas, transfer, unity, wallpapers, web cam, west nile, west nile virus, wrestling

(a)

Tag: mortage jumbo

Bookmarks 1 - 20 of about 160

Viewing Tag: mortage jumbo Clear [x]

Dig Deeper: bad credit mortage, 40 year mortage, calculator interest only mo..., central mortage, foreclosure lists, mortage foreclosure, 30 year fixed mortage, choice homes, country homes, country wide mortage, current mortage rate, current mortage rates, manufactured homes, mortage amortization schedu..., mortage fraud, mortage lending, mortage loan, mortage loan calculator, mortages rates, mortgage calc [view all]

Advice On Second Property Mortgage Offers Posted...
Shared by: jacob m on 8/13/2007 at 11:36 AM - Details
1 save Save
Tags: 15 year mortage, 30 year fixed mortag...,

Why Do I Need A poor Credit Mortgage?
Shared by: jacob m on 8/13/2007 at 11:33 AM - Details
1 save Save
Tags: 30 year fixed mortag..., 30 year

Benefits of an Offset Mortgage
Shared by: jacob m on 8/13/2007 at 11:26 AM - Details
1 save Save
Tags: 15 year mortage, 40 year mortage, 80 20

Pay Option ARMS - What Are They and How Do They ...
Shared by: jacob m on 8/13/2007 at 11:21 AM - Details
1 save Save
Tags: 30 year mortage, 40 year mortage, 50

What do Lenders Consider When Approving poor Cre...
Shared by: jacob m on 8/13/2007 at 11:20 AM - Details
1 save Save
Tags: 2nd mortage, 30 year mortage rate..., abn

Mortgage 3 Day Right of Rescission - Understandi...
Shared by: jacob m on 8/13/2007 at 11:15 AM - Details
1 save Save
Tags: 30 year fixed mortag..., 30 year

Endowment Mortgages Posted By : James Miller
Shared by: jacob m on 8/13/2007 at 11:09 AM - Details
1 save Save
Tags: 2nd mortage, 40 year mortage, abn

Switch your mortgage today Posted By : James Mil...
Shared by: jacob m on 8/13/2007 at 11:08 AM - Details
1 save Save
Tags: 30 year fixed mortag..., 30 year

(b)

Figure 2. An example of tag spam on a popular real-world tagging site captured 13 August 2007. (a) A tag cloud with spam tags pertaining to mortgages. (b) shows some URLs posted with the tag "mortage jumbo," all posted by the same user and with similar names. All posts in (b) point to documents on the same spam site.

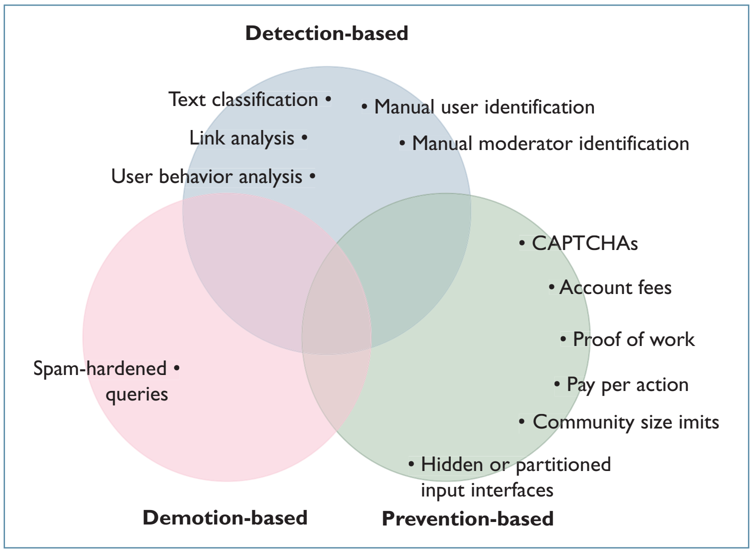


Figure 3. The three main anti-spam strategies: detection, demotion, and prevention. Detection-based strategies attempt to identify spam and remove it or reduce its prominence. Demotion-based strategies attempt to lower the ranking of spam in ordered lists. Prevention-based strategies attempt to make contribution of spam more difficult by changing interfaces or limiting user actions.

Social Bookmarking Example

As users browse the Web, they often locally save visited URLs as bookmarks for later retrieval. Users of a social bookmarking site, however, post them online, optionally with descriptive tags. They can then access these bookmarks from many locations as well as share them with others. They can also query the shared pool of bookmarks for certain statistics, such as the most popular and most recent sets of bookmarks.

In our example of a social bookmarking system (Figure 1), there's only one interaction for content creation, consisting of a single action: a user may post a bookmark. A bookmark consists of a URL and, optionally, a list of tags describing the URL. This information is then displayed through one or more interfaces:

- *user bookmarks* — for a given user *u*, a list of the most recent URLs that user posted;
- *tag bookmarks* — for a given tag *t*, a list of the most recently posted URLs annotated with tag *t*;
- *most recent list* — a list of the most recent URLs posted to the system by any user;
- *most popular list* — a list of the most popular URLs ordered as a function of number of postings and the amount of recent activity for each URL; and
- *tag cloud* — a list of the most commonly occurring tags in the system.

Social bookmarking systems such as del.icio.us, Yahoo! MyWeb, and Furl implement these standard interfaces, although they're by no means the only ones possible. Figure 2 shows two examples of spam, one in the tag cloud interface, and the other in the tag bookmarks interface.

Anti-Spam Strategies

Let's now look at what we believe are the three main anti-spam strategies commonly in practice (Figure 3). For each, we describe the method, note prior work, and then apply the strategy to our social bookmarking example. Figure 4 shows Figure 1's version of the system updated with these strategies.

Identification-Based (Detection)

Identification-based methods proceed in two phases. In the first, likely spam is identified. Users can either manually identify spam, or the system owner can use pattern-based classification. In the second phase, the system interfaces take into account the likely spam and either delete it and then compute their results, or display particular results as likely spam.

For these methods, we can treat the corpus as a set of objects with associated attributes. In email spam, the messages are objects and the headers are

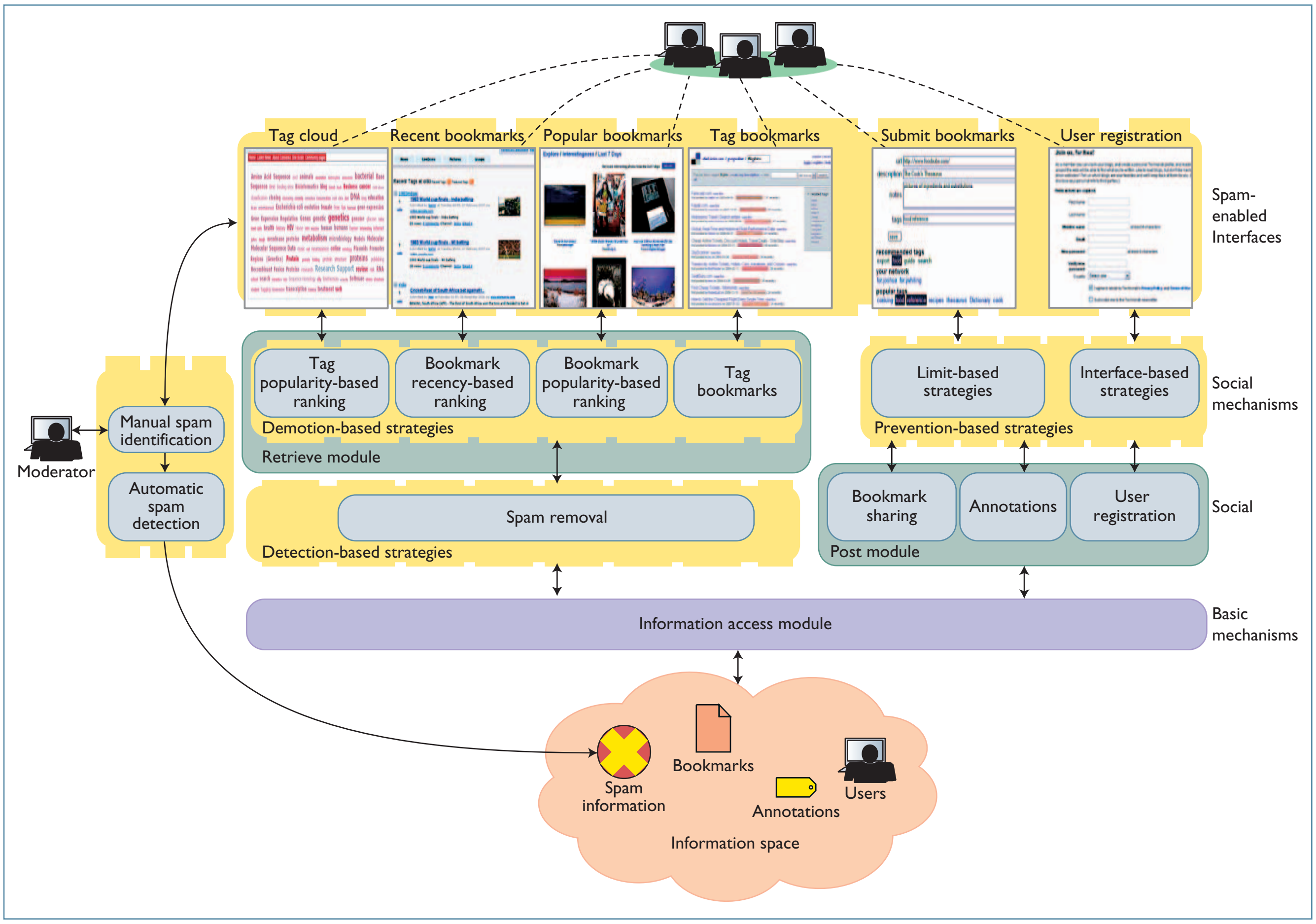


Figure 4. An updated version of Figure 1 with anti-spam strategies applied. Yellow denotes added "spam fighting" functionality. We also label the location at which each strategy (detection, demotion, and prevention) was applied.

attributes. In Web spam, the Web pages are objects, and attributes might be inlinks, outlinks, page content, and various external metadata (such as the speed of a page rank increase or how many domain names resolve to the same IP). Social Web sites will often focus on a few types of objects, such as photos or URLs, and the attributes will then pertain to these objects or their relationships.

With manual identification, users or trusted moderators determine if a given object is spam. This is also usually a prerequisite for automated classification methods, which typically require a training set of objects (classified either as spam or not spam) or a seed set of objects (spam or not spam) from which to propagate trust. Such methods look at patterns of objects or attributes that indicate spam:

- *source analysis* looks at the identities of object contributors;
- *text analysis* looks at words or phrases; and
- *link* or *behavior analysis* looks at networks of objects or users.

This is not an exhaustive list. Each of these identification methods is used in email and on the Web.

Manual spam reporting is common for training email spam filters, and is used by sites such as Wikipedia and Digg. Source analysis occurs in email with whitelists (such as the Habeas Sender Warranted Email service; www.habeas.com) and blacklists (such as SpamCop at www.spamcop.net and the SpamHaus project at www.spamhaus.org) and on the Web with tools like host-based TrustRank.[1] Text analysis in email happens both with terms (especially using Bayesian methods[2]) and with textual corpora of bad messages (such as Vipul's Razor, a community of users who share spam messages [http://razor.sourceforge.net], and other tools designed to share signatures of spam messages[3]). It also happens to a lesser extent on the Web, mainly detecting stylistic elements.[4,5]

Finally, link analysis is very common on the Web[1] but is also used in email social networks.[6]

In our social bookmarking example, we could apply most of these methods. We could provide an interface for users or moderators to label bookmarks as spam. Furthermore, we could analyze the system data to determine if particular IP addresses, users, domains, or text tags are associated with spam. We might also be able to analyze bookmark-sharing mechanisms to determine if some sources or bookmarks are behaving unnaturally. Often, unnatural behavior is a result of automated access to the system or testing of the system for flaws. These methods aren't unique to our example — any social Web site can have an interface to report spam. We can also always perform text analysis on user-generated content, source analysis on IP addresses or user identifiers, and behavior analysis on well-defined user interactions.

Rank-Based (Demotion)

Another strategy for combating spam is to design the system to reduce the prominence of content likely to be spam. Rank-based methods try to provide better results ordering for interfaces that produce lists of results. Ideally, these orderings are both more accurate and more resistant to spam.

Rank-based methods are commonly applied on the Web. Search engines only return the top *k* results, where *k* is much smaller than the size of the Web as a whole. Often improving the top *k* results will lead to a reduced perception of spam, even though technically the spam remains deep in the ranked results. Rather than directly removing spam, for example, trust-based methods like TrustRank[1] can reduce the rank of likely spam. However, rank-based methods are more difficult to apply to email because messages are almost always ordered by time.

In our social bookmarking example, we could apply a rank-based method by improving the algorithm that determines the most popular list. For instance, we might change the ordering for a Web site with a geographically diverse user base to be a function of both bookmark occurrences and bookmark contributors' geographical location. This function might rank a URL that's bookmarked by fewer users but with geographically disparate IP addresses higher than a URL with more bookmarks but from a more geographically concentrated set of users. This would lead to a broader notion of popularity and make it more difficult to create spam in the most popular list. However, it might be more difficult to apply rank-based methods to time-ranked interfaces, such as the most recent list. More generally, most social Web sites can use rank-based methods to improve the resistance of targeted result-based interfaces.

Interface- or Limit-Based (Prevention)

Prevention methods try to make contributing spam content difficult; system designers can achieve this by making some details of the system secret or by limiting automated interaction. Two common forms of prevention exist — interface- and limit-based. Interface-based methods attempt to either hide or restrict access to actions that let users contribute content. Limit-based methods try to limit the resources needed for user actions in the system so that economic considerations or social norms apply. Commonly, limit-based systems will limit the number of new user accounts or how often users can perform certain actions. These limits can either be hard-number limits or implicit limits imposed by charging money for certain interactions.

Interface-based methods are common both on the Web and with email. On the Web, we can use CAPTCHAs[7] (a type of challenge-response test that stands for completely automated public Turing test to tell computers and humans apart; www.captcha.net) to prevent programmatic access to a Web site. Disposable email addresses (for example, Spam Gourmet and other providers of one-time or limited-use email addresses), challenge-response messages,[8,9] and graylisting (essentially asking the email sender to "come back later")[10] are all common interface-based methods for making programmatic sending of email messages difficult. Limit-based methods have been more commonly proposed than implemented, although examples abound of each. With email, there have been various proposals for "stamps," which cost either computational time (such as proofs-of-work system HashCash [www.hashcash.org] and Microsoft's PennyBlack [http://research.microsoft.com/research/sv/PennyBlack/], or other work[11]) or real money.[12] On the Web, although there's no set interaction cost, certain resources are more expensive than others. For instance, `*.edu` domains are difficult and expensive for a search engine spammer to acquire, whereas `*.info` domains are much less so. Various Web communities, such as Kuro5hin (www.kuro5hin.org), have experimented with ceasing creation of new accounts in order to strengthen existing user-moderation systems.

In the social bookmarking example, we could apply any of these methods. We could apply interface-based methods by using CAPTCHAs to prevent automated account creation or automated bookmark posting. Aside from blocking automated input, we might also be able to hide or reduce the value of certain interfaces. For instance, if the most popular list were a prominent interface, we could personalize it on a per-user basis, making it more difficult to reliably spam. (Rather than having a single list of results to penetrate, the attacker would then need to figure out how to penetrate a large subset of users' personalized lists. Having personalized lists increases both the difficulty of success and the difficulty for the attacker to determine that they were successful.) We could apply limit-based methods by making users who register for an account or post a bookmark either pay a small fee or compute a proof-of-work. In fact, a HashCash implementation already exists that's intended to prevent blog comment spam. We could also restrict the creation of new user accounts or limit how many bookmarks could be posted per day. Regardless of the Web site type, there's almost certainly a monetary, computational, or social cost at which spam creation becomes undesirable. However, this cost could drive away legitimate users first.

Evaluation

Ideally, a formal evaluation of the strategies presented here should take place before deploying them in a real social system. Such systems are large and complicated, and users might react negatively to rapid or negatively perceived changes. More formal models for evaluation can prevent system owners from investing resources in implementing strategies that could turn out to be inadequate in practice.

Evaluating the effectiveness of a set of spam countermeasures requires two ingredients: spam models and spam metrics. A *spam model* captures whether content in the system is classified as spam or not spam. A *spam metric* provides a quantitative assessment of how spam affects a particular interface. But deriving good spam models and metrics is complicated by two factors:

- *Subjectivity of spam.* The notion of "spam" content varies from person to person. Whereas one person might consider the tag "ugly" on a photo of Aunt Thelma to be inappropriate, another might think it's perfect! Consequently, the notion of a "malicious" tag is very subjective. Most people would agree that some behaviors are inappropriate, but precisely defining such behaviors isn't easy.
- *Malicious users.* Malicious users can mount several different attacks on a social system. If they know that there's a limit on how much content they can submit, for instance, they can sign up using different identities. If they know that moderators will block users who post spam, they might try to disguise their attacks by contributing a fraction of "good" content. What malicious users do depends on their sophistication, their goals, and on whether they collude. Because malicious users are a moving target, it's hard to know what they'll do next.

Given these difficulties, we look first at possible approaches to spam modeling.

Spam Models

We can model spam in a social system using two approaches: either define an ideal model by making a set of assumptions (a synthetic spam model) or build a model based on real data observation (a trace-driven spam model). Both models lead to labeled data sets for evaluating spam-fighting techniques.

A *synthetic spam model* requires a representation of a social system and of user interactions therein. For instance, our social bookmarking system might be represented by four values:

- $\mathcal{O}$ – the set of system objects, in particular URLs;
- $\mathcal{U}$ – the set of system users;
- $\mathcal{T}$ – the set of system tags; and
- $\mathcal{P}$ – a list of tuples $(o_i, u_j, t_k, ..., t_l)$, each of which denotes that user u_j posted object o_i with tags t_k through t_l.

This representation, when generalized to allow arbitrary object types, is the model we used in recent work.[13] To model user interactions in the system, we also need to define spam content and malicious behavior. In our example, we could assume that each URL has a set of tags that correctly describe it and that malicious behavior consists of selecting an incorrect tag for a particular object. For instance, the URL www.time.gov (a Web site providing the official US time) might appropriately be tagged "time" or "US." All other tags (such as "cat," "Indonesia") are incorrect (spam); there are no ambiguous tags. Once we have defined the system's representation and interactions within it, we can use our model to generate synthetic data. However, we might also want to parameterize the model with variables like the distributions that are used to select tags, queries, and objects. Such parameterization lets us capture how spam-fighting techniques will cope with different situations and user behaviors. However, even though we know a priori which content is spam and what policy malicious users will follow, our spam-fighting techniques aren't aware of this information. Only when it's time to evaluate how successful the technique was do we look at which tags were correct and which were spam.

A *trace-driven spam model* starts with content available in the system or in a publicly available test collection. It then explicitly requires some user or expert input to label spam content; this could be a set of positive and negative examples of spam content or it might have been distilled from many users' aggregate votes. Researchers, system designers, and others can then use this set to evaluate spam-fighting schemes, such as how effective the schemes are at classifying content as spam or demoting it in various user views. Practitioners often use trace-driven spam models for evaluating email and Web spam-fighting methods.[14]

In practice, trace-driven spam models have usually been preferred because they can produce more realistic results. However, they might under-represent content due to sample bias or topic drift in the underlying collection, and they're sometimes time-consuming to create. These weaknesses are more pronounced for social Web sites. Varying modes of interaction and differing subject matter mean that each site (or type of site) potentially might need a different trace-driven spam model. Social Web sites also tend to experience rapid growth, which could promote topic drift as the user base changes. For these reasons, we believe that synthetic models might be more practical for social Web sites, even though we must be more careful in interpreting them.

Spam Metrics

Spam metrics quantify the impact of spam on an interface or the effectiveness of a spam-fighting method at protecting a particular interface. We will describe two basic types of metrics; variations exist but are outside the scope of this article.

One approach is to view the spam problem as a classification problem: given a list of objects, partition it into two lists, one list of spam content and one list of non-spam content. In this case, we can define spam metrics based on the traditional concepts of precision and recall:

- *Precision.* Of the good (or bad) content that we acted on (identified, changed the rank of, or prevented) as good (or bad), what percentage was actually good (or bad)?
- *Recall.* Of the good (or bad) content in the system, what percentage did we act upon as good (or bad)?

Another approach is to view the spam problem as a ranking problem: given a list of objects, compute a rank for each object. This makes the most sense for evaluating demotion-based strategies. In this case, a spam metric could take into account both the number of spam objects and their position in the list. For instance, the SpamFactor metric measures the spam for a ranked set of n results returned for a query tag t in a tagging system.[13] SpamFactor increases as more spam is present in the results, and as that spam is featured more prominently. SpamFactor is computed as

$$SpamFactor(n,t) = \frac{\sum_{i=1}^{n} w(o_i,t) * \frac{1}{i}}{\sum_{i=1}^{n} \frac{1}{i}},$$

where

$$w(o_i,t) = \begin{Bmatrix} 1 \text{ if } t \text{ is a bad tag for } o_i \\ 0 \text{ if } t \text{ is a good tag for } o_i \end{Bmatrix}.$$

Other information retrieval metrics, such as mean average precision (MAP), can also be adapted to evaluate spam's impact on ranked retrieval. Of course, in a real-world system, spam metrics are only one class in a variety of metrics that a system designer could use to improve ranked results.

Evaluation Scenario: Tagging Systems

To illustrate how we can evaluate spam-fighting techniques, and what we can learn from such evaluations, we briefly describe an evaluation we performed. The evaluation uses a very simple synthetic model, but in spite of its simplicity, we believe it can yield useful insights. Details and full results can be found elsewhere.[13]

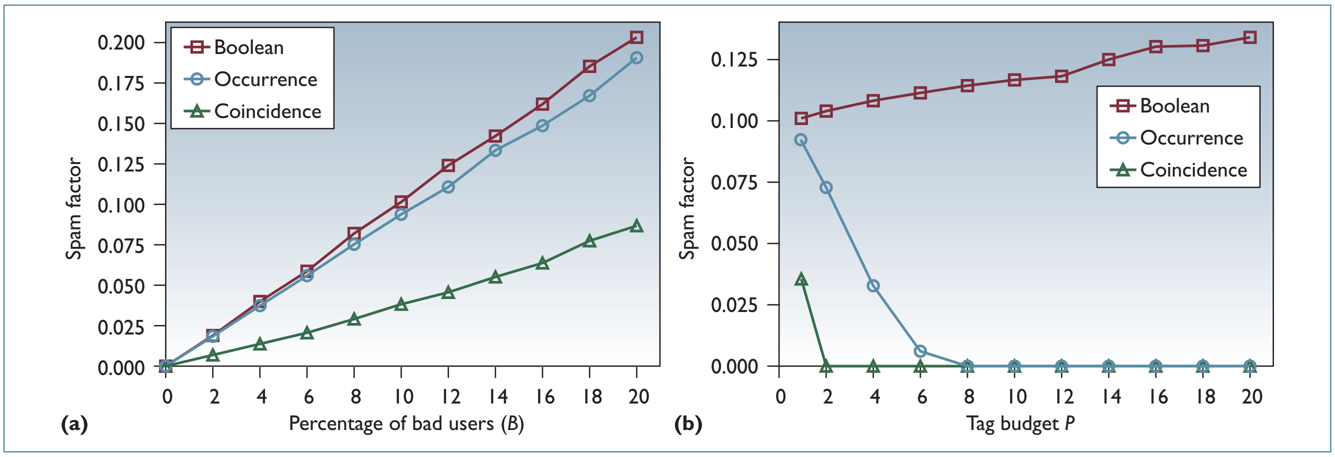


Figure 5. Illustrations of varying different aspects of a synthetic model and their effects on spam. (a) The effect of a varying percentage of bad users on a tagging system, vs. (b) the effect of a varying tag budget for each user.

We focused on a tagging system in which users annotate objects (URLs, photographs, and the like). As described earlier, we generated synthetic objects, each with a set of "good" tags. As a starting point, we considered the following user models: when a good user tags object o_i, he or she selects at random from among o_i's good tags; when a bad user tags o_i, he or she selects at random among tags that aren't good for o_i.

The bad user model captures the behavior of a "lousy but not malicious" user. In other work, we also considered bad users who are malicious and collude.[13] For the evaluation, we considered 10,000 objects and 1,000 users in the system and a set of 500 tags that users can apply to their postings. Each user can contribute 10 postings.

We studied the effect of users' tagging behavior for single-tag queries. Given a query containing a tag t, the system returns a ranked list of objects that contain t. We considered the following three search schemes:

- *Boolean model*, which randomly orders objects that match the query tag t in the results.
- *Occurrence-based model*, which ranks each object o_i by counting the number of tuples in $\mathcal{P}$ with object o_i and tag t.
- *Coincidence-based model*, which takes into account users' reliability. We can measure reliability in this way: user u_i is considered more reliable than user u_j if u_i's tags more often coincide with other users' tags compared to u_j's tags. (That is, u_i is more often in agreement with its peers.) To measure how reliable a user is, we define the *coincidence factor* $c(u)$ of a user u as follows:

$$c(u) = \sum_{o,t:\exists \mathcal{P}(u,o,t)} \sum_{\substack{u_i \in \mathcal{U} \\ u_i \neq u}} \left|\mathcal{P}(u_i,o,t)\right|,$$

where $\mathcal{P}(u_i, o, t)$ represents the set of postings by user u_i that associate o with t. Given a query tag t, we can account for coincidence factors for ranking documents returned for a specific query tag. Then, the rank of a document d with respect to t can be computed as follows:

$$rank(d,t) = \frac{\sum_{\forall u \in users(d,t)} c(u)}{c_o},$$

where $users(d, t)$ is the set of users that have assigned t to d, and c_o is the sum of coincidence measures of all users. The latter is used for normalization purposes so that a rank ranges from 0 to 1.

To illustrate what researchers and system designers can do with such a model, we consider here two of the questions studied in other work:[13] Does leveraging social knowledge (like tag coincidences) help in fighting spam? Does limiting the number of tags per user (a tag budget) help combat spam?

Figure 5a illustrates the effect that varying the percentage of bad users in the system has on the SpamFactor value of tag search results. SpamFactor grows linearly because more bad users contribute more bad postings. Relying on the number of occurrences of the query tag in an object's postings generates results that are marginally better than randomly selected objects matching the query tag. Using tag coincidences works substantially better, cutting SpamFactor by a factor of two. The reason behind this improvement is that more informed decisions regarding which objects to select are possible when relying not only on the fact that an object is associated with a tag, but also on the number and reliability of users who have claimed so. Consequently, leveraging social knowledge can help fight spam, at least with "lousy" bad users who do not collude. Of course, when bad users proliferate, all countermeasures will become gradually less effective.

Figure 5b shows how tag search results are affected by varying the number of tags allowed per user (tag budget) from 2 to 200 for a moderate bad user population (10 percent of the overall user population). We observe that when users provide more postings in the system, duplicate good postings accumulate, helping occurrence-based searches to generate results that are increasingly better than Boolean results. Coincidence-based results are even better because as users contribute more tags, tag coincidences will occur more often, thus the system can identify reliable users more easily. Consequently, active good users are beneficial for tag searches because they can help promote good objects. Thus, forcing a tag budget per user to limit the negative impact of malicious users also constrains the positive impact on the system due to good user activity.

If email and Web spam are any lesson, what we're witnessing today is only the first round in what will likely be an unending conflict. Currently, much of the spam on social Web sites can probably be detected by the off-the-shelf methods surveyed so far. Over time, we anticipate that anti-spam techniques for social Web sites will diverge from email and the Web. Spam contributors will start to create strategies based on the varied set of interactions that social Web sites provide, such as voting and spam reporting, as well as currently deployed countermeasures, and the next round will begin.

In the meantime, we have surveyed pre-existing solutions to the problem of spam on social Web sites. In many ways, these solutions work even better for social Web sites than for the domains for which they were intended. In particular, detailed logs of user interaction can be kept, greatly aiding identification-based methods. The single controlling entity of a social Web site can also force interface and limit-based solutions that weren't possible with the widely distributed email and Web systems. On the other hand, mandating interface and limit-based solutions could also alienate inconvenienced users or even be counterproductive. Consequently, evaluation of anti-spam strategies should take place, even before deploying them in a real social system, in order to gain insight into their merits and foresee possible shortcomings. For this purpose, we need to derive appropriate spam models and metrics, which, as we have seen, is not trivial. Moreover, our evaluation metrics might always be one step behind as more sophisticated strategies and evaluation metrics have a habit of breeding more sophisticated adversaries.

Beyond these methods' raw effectiveness, we believe that one of the most interesting challenges in this area is determining the mix of countermeasures that users find most appropriate. This is also likely to be community-specific. Some communities are likely to favor anti-spam, which requires manual human labor, because these solutions are transparent and create user involvement. However, others might perceive behind-the-scenes algorithms or trusted moderators as less likely to be biased. Figuring out the proper balance of countermeasures, together with the potential for interesting new rank-based methods and evaluation metrics for a multitude of new interfaces, leads us to believe this will be a fertile research area in the future.

References

1. Z. Gyöngyi, H. Garcia-Molina, and J. Pedersen, "Combating Web Spam with TrustRank," *Proc. 30th Very Large Databases Conf.*, 2004, pp. 576–587; http://dblp.uni-trier.de/db/conf/vldb/vldb2004.html#GyongyiGP04.
2. P. Graham, "A Plan for Spam," Aug. 2002; www.paulgraham.com/spam.html.
3. A. Gray and M. Haahr, "Personalised, Collaborative Spam Filtering," *Proc. 1st Conf. Email and Anti-Spam* (CEAS 04), 2004; www.ceas.cc/papers-2004/132.pdf.
4. T. Urvoy, T. Lavergne, and P. Filoche, "Tracking Web Spam with Hidden Style Similarity," *Proc. 2nd Int'l Workshop on Adversarial Information Retrieval on the Web* (AIRWeb 06), 2006; http://airweb.cse.lehigh.edu/2006/proceedings.pdf.
5. G. Mishne, D. Carmel, and R. Lempel, "Blocking Blog Spam with Language Model Disagreement," *Proc. 1st Int'l Workshop on Adversarial Information Retrieval on the Web* (AIRWeb 05), 2005; http://airweb.cse.lehigh.edu/2005/#proceedings.
6. L.H. Gomes et al., "Comparative Graph Theoretical Characterization of Networks of Spam," *Proc. 2nd Conf. Email and Anti-Spam* (CEAS 05), 2005; www.ceas.cc.
7. L. von Ahn et al., "CAPTCHA: Using Hard AI Problems for Security," *Proc. Eurocrypt*, 2003, pp. 294–311; citeseer.ist.

psu.edu/vonahn03captcha.html.

8. B. Templeton, “Proper Principles for Challenge/Response Anti-Spam Systems,” Oct. 2007; www.templetons.com/brad/spam/challengeresponse.html.
9. E. Felten, “A Challenging Response to Challenge-Response,” *Freedom to Tinker*, 19 May 2003; www.freedom-to-tinker.com/archives/000389.html.
10. J.R. Levine, “Experiences with Greylisting,” *Proc. 2nd Conf. Email and Anti-Spam* (CEAS 05), 2005; www.ceas.cc.
11. C. Dwork and M. Naor, “Pricing via Processing or Combating Junk Mail,” *Proc. 12th Ann. Int’l Cryptology Conf. Advances in Cryptology* (CRYPTO 92), Springer-Verlag, 1993, pp. 139–147.
12. B. Templeton, *E-stamps*, Oct. 2007; www.templetons.com/brad/spam/estamps.html,.
13. G. Koutrika et al., “Combating Spam in Tagging Systems,” *Proc. 3rd Int’l Workshop Adversarial Information Retrieval on the Web* (AIRWeb 07), 2007; http:/dbpubs.stanford.edu/pub/2007-11.
14. T. Lynam and G. Cromack, “TREC Spam Filter Evaluation Toolkit,” Oct. 2007; http://plg.uwaterloo.ca/~gvcormac/treccorpus/.

Paul Heymann is a PhD candidate in Stanford University’s Department of Computer Science. His research focuses on collaborative tagging and its potential impact on traditional information retrieval systems. Heymann has a dual bachelors degree in computer science and philosophy from Duke University. Contact him at heymann@stanford.edu.

Georgia Koutrika is a postdoctoral researcher at Stanford University’s Infolab. Her research interests include social systems, social spam, keyword searching, database and Web personalization, digital libraries, user modeling and profiling, and query processing and optimization. Koutrika has a PhD, an MSc in advanced information systems, and a BSc in informatics from the University of Athens’ Department of Informatics and Telecommunications. Contact her at koutrika@stanford.edu.

Hector Garcia-Molina is the Leonard Bosack and Sandra Lerner Professor in the departments of computer science and electrical engineering at Stanford University. His research interests include distributed computing systems, digital libraries, and database systems. Garcia-Molina has a BS in electrical engineering from the Instituto Tecnologico de Monterrey, Mexico, an MS in electrical engineering, and a PhD in computer science from Stanford University. He is a fellow of the ACM and of the American Academy of Arts and Sciences and a member of the National Academy of Engineering; he received the 1999 ACM SIGMOD Innovations Award. Contact him at hector@cs.stanford.edu.

IEEE computer society

PURPOSE: The IEEE Computer Society is the world’s largest association of computing professionals and is the leading provider of technical information in the field.
MEMBERSHIP: Members receive the monthly magazine *Computer*, discounts, and opportunities to serve (all activities are led by volunteer members). Membership is open to all IEEE members, affiliate society members, and others interested in the computer field.
COMPUTER SOCIETY WEB SITE: www.computer.org
OMBUDSMAN: Email help@computer.org.

Next Board Meeting: 16 May 2008, Las Vegas, NV, USA

EXECUTIVE COMMITTEE

President: Michael R. Williams*
President-Elect: Rangachar Kasturi;* **Past President:** Deborah M. Cooper;* **VP, Conferences and Tutorials:** Susan K. (Kathy) Land (1ST VP);* **VP, Electronic Products and Services:** Sorel Reisman (2ND VP);* **VP, Chapters Activities:** Antonio Doria;* **VP, Educational Activities:** Stephen B. Seidman;† **VP, Publications:** Jon G. Rokne;† **VP, Standards Activities:** John Walz;† **VP, Technical Activities:** Stephanie M. White;* **Secretary:** Christina M. Schober;* **Treasurer:** Michel Israel;† **2006–2007 IEEE Division V Director:** Oscar N. Garcia;† **2007–2008 IEEE Division VIII Director:** Thomas W. Williams;† **2007 IEEE Division V Director-Elect:** Deborah M. Cooper;* ***Computer* Editor in Chief:** Carl K. Chang;† **Executive Director:** Angela R. Burgess†

** voting member of the Board of Governors   † nonvoting member of the Board of Governors*

BOARD OF GOVERNORS

Term Expiring 2007: Jean M. Bacon, George V. Cybenko, Antonio Doria, Richard A. Kemmerer, Itaru Mimura, Brian M. O’Connell, Christina M. Schober
Term Expiring 2008: Richard H. Eckhouse, James D. Isaak, James W. Moore, Gary McGraw, Robert H. Sloan, Makoto Takizawa, Stephanie M. White
Term Expiring 2009: Van L. Eden, Robert Dupuis, Frank E. Ferrante, Roger U. Fujii, Ann Q. Gates, Juan E. Gilbert, Don F. Shafer

EXECUTIVE STAFF

Executive Director: Angela R. Burgess; **Associate Executive Director:** Anne Marie Kelly; **Associate Publisher:** Dick Price; **Director, Administration:** Violet S. Doan; **Director, Finance and Accounting:** John Miller

COMPUTER SOCIETY OFFICES

Washington Office. 1730 Massachusetts Ave. NW, Washington, DC 20036-1992
Phone: +1 202 371 0101 • Fax: +1 202 728 9614
Email: hq.ofc@computer.org
Los Alamitos Office. 10662 Los Vaqueros Circle, Los Alamitos, CA 90720-1314
Phone: +1 714 821 8380 • **Email: help@computer.org**
Membership and Publication Orders:
Phone: +1 800 272 6657 • Fax: +1 714 821 4641
Email: help@computer.org
Asia/Pacific Office. Watanabe Building, 1-4-2 Minami-Aoyama, Minato-ku, Tokyo 107-0062, Japan
Phone: +81 3 3408 3118 • Fax: +81 3 3408 3553
Email: tokyo.ofc@computer.org

IEEE OFFICERS

President: Leah H. Jamieson; **President-Elect:** Lewis Terman; **Past President:** Michael R. Lightner; **Executive Director & COO:** Jeffry W. Raynes; **Secretary:** Celia Desmond; **Treasurer:** David Green; **VP, Educational Activities:** Moshe Kam; **VP, Publication Services and Products:** John Baillieul; **VP, Regional Activities:** Pedro Ray; **President, Standards Association:** George W. Arnold; **VP, Technical Activities:** Peter Staecker; **IEEE Division V Director:** Oscar N. Garcia; **IEEE Division VIII Director:** Thomas W. Williams; **President, IEEE-USA:** John W. Meredith, P.E.

revised 11 Oct. 2007

IEEE