SpamClean: Towards Spam-free Tagging Systems

Ennan Zhai[†,§], Huiping Sun[†,§], Sihan Qing[†], Zhong Chen[†,§]

[†]School of Software and Microelectronics, Peking University, China
Email: zhaien@infosec.pku.edu.cn, {sunhp, qsihan, chen}@ss.pku.edu.cn
[§]Key Laboratory of High Confidence Software Technologies, Ministry of Education, China

***Abstract*—Tagging systems are known to be particularly vulnerable to *tag spam*. This paper introduces SpamClean, a novel *social experience-based* scheme, and presents the performance of SpamClean to defend against the tag spam in tagging systems. We first propose a novel mechanism based on cosine technique to compute the correlations between the client and other users in the system, and look the correlations as the experiences of the client with respect to other users. The client ranks each tag search result based on the average of experiences of the client with respect to all the owners of this result. To obtain higher quality search results, we propose *socially-enhanced mechanism* — using the friend-relationships, the social nature of tagging systems, to enhance SpamClean. This is based on considering that the client's social friends can share their previous experiences and help improve both the performance and convergence of SpamClean. Finally, the experimental results illustrate that SpamClean can effectively defend against tag spam and work better than the existing search models in the current tagging systems.**

I. Introduction

Tagging services in social networks, e.g., Flickr [1], Del.icio.us [2], YouTube [3], have grown in significance on the Internet in terms of the number of participating users. In a typical tagging system, each specific *resource* (photo, URL) is annotated with some *tags*. The users, who have annotated a specific resource with some tags, are called the *resource annotators*, and the relation $\langle tag, resource\rangle$ that annotates a resource with a tag is called an *annotation*, which maintains the association between the tag and resource. When the *client* issues a tag search, the system returns resources associated with this tag. Then, the client may collect some of them, and annotates these resources with some tags. Many recent studies indicated that the tagging systems are vulnerable to *tag spam*: the erroneous or misleading tags that are generated by some malicious users to confuse the normal participants in the system [4], [5]. For instance, some attackers may repeatedly annotate some videos in YouTube with the erroneous tags, so that the normal users, without sufficient knowledge about other participants, may be misled to open an undesirable movie.

In this paper, we propose SpamClean — a novel social experience-based scheme towards *spam-free* and *personalized* tag search. SpamClean encompasses two key mechanisms: *experience mechanism* and *socially-enhanced mechanism*:

- **Experience Mechanism:** SpamClean enables each client to assign each of other users in the system with a personalized *experience*, so that, for a typical tag search, the search results can be ranked by the average of the client's experiences with respect to the annotators of each search result. SpamClean client computes the statistical correlation of annotations between the client and annotator, as the experiences with respect to the annotator, and the calculation of correlation is based on the cosine technique. The fundamental insight driving our work is that the users, who have the similar annotations with the client, can provide the more reliable search results. Meanwhile, the scheme of computing the statistical correlation of annotations also provides a strong incentive for each client to participate in correctly annotating, since the users who do not annotate correctly and actively will find the quality of the search results they compute noticeably degraded.
- **Socially-enhanced Mechanism:** Computing experiences can work well for the users who have the common annotations with the client, but will not discover relationships between the client and user with few common interests. To address this problem and provide the more effective mechanism, we utilize friend-relationships, the social nature of tagging systems, to make SpamClean more robust to the tag spam. In current tagging systems, the friends are either acquaintances in reality, or those online friends recognized in social networks, such as Facebook. Therefore, these reliable companions can provide many references, based on their previous experiences, to enhance the performance of SpamClean.

We conducted simulation studies on several synthetic centralized systems with different user, tag, resource and execution models, and compared SpamClean with the existing tag search models, e.g., *Boolean* [6], *Occurrence* [7], and *Coincidence* [5]. The evaluation results show that SpamClean can defend against tag spam from various attackers more effectively than the existing search models. In addition, SpamClean can easily be extended to incorporate with the existing defense mechanism against Sybil [8] and Denial-of-Service (DoS) attacks — the acknowledged threat in the tagging systems.

To the best of our knowledge, there is not any study, using the method similar with the social experience, on defending against tag spam. Our research contributions are as follows:

- We propose SpamClean, a novel social experience-based scheme against tag spam in the tagging systems.
- We are the first to enhance the robust of defense mechanism against tag spam with the friend-relationships — the social nature of tagging systems.

- We compare SpamClean with the various search models under the different threat models, and present the effectiveness of SpamClean against tag spam.

Outline: The rest of this paper is organized as follows. We present the related work in Section II. Section III describes the design rationale of SpamClean. Then, the simulation methodology and evaluation results are discussed in Section IV. Finally, we conclude with a discussion of incorporating SpamClean with the existing defenses against Sybil and DoS attacks in Section V.

II. Related Work

Currently, there are some mechanisms proposed to address tag spam in the tagging systems. In general, these mechanisms can be grouped into three categories [4]: *detection-based* mechanisms, *demotion-based* mechanisms and *interface-based* mechanisms.

Detection-based Mechanisms: This category of mechanisms are mainly based on the principles of statistical analysis and machine learning. The study in [9] investigated the usefulness of different machine learning algorithms on the calculations of features. Besides, based on determining relevant features to tagging systems, the author transferred those approaches to identify those spammers. The studies in [10], [11] proposed the algorithms which could detect the tag spam through training and learning. Based on the notion that similar users and annotations tend to use the same language, the study in [12] introduced *language model* to address the problem of spam in tagging systems.

Demotion-based Mechanisms: In this category of anti-spam mechanisms, the algorithms can return the most popular list and degrade the rank of spam to the end of search results. The study in [13] took into account spam by proposing a credibility score for each user based on the quality of the tags contributed by the user. Studies in [4], [5] proposed a simplified model of tagging behavior in tagging systems and compared different ranking methods for tag-based searching.

Interface-based Mechanisms: These methods attempt to hide or restrict access to actions that make users contribute content. CAPTCHAs [14] can be used to prevent automated account creation or automated tag spam annotation.

III. Design Rationale of SpamClean

In this section, we describe two key mechanisms of SpamClean, and the solution on the encountered practical issue. First, we describe the *experience mechanism* in Section III-A, and the *socially-enhanced mechanism* in Section III-B. Then, we present the solution to the encountered practical issue in Section III-C.

A. Experience Mechanism

In SpamClean, if the client wants to acquire a specific resource R, he will issue a tag search t to the system, and then the system will present the *search result pages*, including the matching resources, to the client. In the search result pages, the search results can be ranked by the average of the client's experiences with respect to the annotators of each search result (resource). Therefore, the ranking score of each result may be interpreted as a personalized estimate of the authenticity of the annotation $\langle t, R\rangle$, and help the client make a decision to choose the resources.

Calculation of Experience: In SpamClean, client A assigns a personalized experience to each other participants in the system. Specifically, client A uses $E_{A,B}$ to denote the experience with respect to the user B. The study in [15] demonstrated that the statistical correlation of annotations between A and B can reflect precisely the reliability of user B in A's view. Therefore, SpamClean proposes the cosine technique to compute the statistical correlation of annotations between A and B, and uses this correlation as the experience $E_{A,B}$. We define V_i as the user i's *annotation vector* which is comprised of all the annotation of the user i, and VM_i denotes *vector measure* of V_i. The details of computing vector measure and the similarity between VM_i and VM_j, $SVM_{i,j}$, are defined as in Equation 1. The equation that computes the correlation ($E_{A,B}$) between the client A and the user B is as follows:

$$\begin{cases} E_{A,B} = SVM_{A,B}/(\sqrt{VM_A}\cdot\sqrt{VM_B}) \\ VM_A = \sum_{r_j\in R}(\sum_{t_i\in T_{A(r_j)}} |N(t_i,r_j)|)^2 \\ VM_B = \sum_{r_j\in R}(\sum_{t_i\in T_{B(r_j)}} |N(t_i,r_j)|)^2 \\ SVM_{A,B} = \sum_{r_j\in R}(\sum_{t_i\in C_{r_j}} |N(t_i,r_j)|)^2 \end{cases} \tag{1}$$

where

- R: The set of resources shared by A and B in common.
- r_j: The jth resource of the common resource set R.
- C_{r_j}: The set of the tags annotated by A and B in common to the resource r_j.
- t_i: The ith tag of the tag set.
- $T_{x(r_j)}$: The set of tags annotated by the user x to the resource r_j.
- $N(t_i, r_j)$: The set of annotation that annotated r_j with t_i.
- $|N(t_i, r_j)|$: The size of $N(t_i, r_j)$.

The range of $E_{A,B}$ is $[0, 1]$, and higher value indicates that the user B is more reliable for A and small $E_{A,B}$ indicates either the user B is unreliable for A or the lack of any significant relationship between A and B.

Incentive: We also notice that the client can only compute the accurate and significant correlations for others, if he has himself cast a sufficient high ratio of the overlapping annotations. Overlapping annotations mean that both the resources and the tags of two annotations are the same. This restriction provides a strong incentive for users to participate in annotating, since the users who do not annotate correctly and actively will find the quality of the estimate they compute noticeably degraded. Indeed, a client can still benefit from

SpamClean by annotating honestly but inactively, suppressing the sharing and dissemination of erroneous tags to the system.

B. Socially-enhanced Mechanism

In the practical applications, experience mechanism can not address one typical issue that lack of overlapping annotations. For example, due to the absence of the common resources with the strange user B, the client A is unable to obtain the ideal experience with respect to the user B, through computing the statistical correlation. Another example, due to the insufficient annotating and collecting behaviors, a newcomer probably does not have the sufficient annotations to compute the exact experiences with respect to most users in the system. Thus, he may be a victim. To address the above problem, we explore the solution through utilizing the characteristic of tagging systems. Considering about the social nature of tagging systems, we utilize the friend-relationships of tagging systems to enhance the performance of SpamClean.

Establish Friend-relationships: In socially-enhanced SpamClean, the client may have many friends and stores their information in the client's *friend list*. The client can establish friend-relationships with the users who are either his acquaintances in reality or those online friends recognized in social networks, such as Facebook. SpamClean utilizes the friend-relationships based on the fundamental fact that the friends are more reliable than those unknown users in the tagging system. Note that the client's friends may be malicious or compromised, and thus we will present an approach to addresses this practical issue, in Section III-C.

Enhanced Mechanism: Due to utilizing the friend-relationships in the tagging systems, the client can address the problem — the lack of overlapping annotations (weak or no correlation), with the approach which is similar with the *majority voting*. After the client ranks his search results, he will find out all the resources which have the ranking scores lower than 0.5 in his search results, and then the client sends all his friends some *enhanced requests* which demand these friends to compute ranking scores for those resources. For a result, if more than half of the friends return the ranking scores of the above resource higher than 0.5, SpamClean will re-locate the position of the resource in the client's result page according to the new ranking score by computing the average of these friends' returned scores higher than 0.5. Otherwise, the client maintains the original rank unchanged.

For instance, we assume that the resource $sea.jpg$ has the ranking score 0.2 (lower than 0.5) in the client's result page, and the client has three friends in the system. Using socially-enhanced mechanism, the client obtains ranking scores, 0.9, 0.7 and 0.3, from his three friends, and thus he should replace $sea.jpg$'s original ranking score 0.2 with $(0.9 + 0.7)/2 = 0.8$. Therefore, the position of $sea.jpg$ in the client's result page is changed. Another example, the ranking scores from the client's three friends are 0.6, 0.1 and 0.3 respectively. Because lower than half of the friends provide the ranking scores higher than 0.5, the ranking score of the resource $sea.jpg$ remains unchanged.

The design of enhanced mechanism is based on the following considerations. The fact that resource has a low ranking score in the client's search result pages is probably caused by the reason that the client has no sufficient correlation with the annotators of the resource. Therefore, the client chooses to seek help from his friends to compute the ranking score of the resource again. If the newly computed ranking score is higher than 0.5, we consider this value a relatively reliable score, so the client will re-locate the position of the resource in his result page; otherwise, the client will maintain the position of the resource unchanged.

C. Practical Issue

To avoid the harms incurred by malicious friends in the practical applications, SpamClean proposes the reliability degree of the client A with the friend f, $R_{A,f}$, for helping the client find out the unreliable friends. The initial value of $R_{A,f}$ is set to 1, since the friends are all considered to be trustworthy in the beginning. Here, the client may evaluates the search result with $+1$ or -1, where $+1$ denotes the satisfied result, and -1 represents the erroneous annotation. Once the client evaluates -1 to an annotation in the search result pages, SpamClean examines whether there are some of the client's friends who publish this erroneous annotation. If some friends indeed publish this erroneous result, SpamClean client will decreases the reliability degrees with them. When the reliability degree of the client's friend drops to below 0, the client will not send the enhanced request used in the socially-enhanced mechanism to this friend, since this value represents considered weak or no reliability. Therefore, the malicious friends can not continue to provide the erroneous search results. Meanwhile, if the client casts $+1$ to a search result, SpamClean also examines whether some client's friends annotate the result with the current tag. This mechanism ensures the friends who have the low reliability degrees can recover their reliability degree. The specific algorithm is as follows:

$$R_{A,f} = \begin{cases} \max(-1, R_{A,f} - \beta n^2) & \text{if result is erroneous} \\ \min(1, R_{A,f} + \alpha) & \text{otherwise} \end{cases} \tag{2}$$

where

- n: the number of consecutive discoveries of erroneous annotations from friend f (including the last one).
- β: the penalty factor given to friend f for each erroneous annotation result evaluation.
- α: the recompense given to friend f for each correct annotation result evaluation.

Note that, in this case, the reliability degree of A with respect to f decreases faster than it increases. Aiming at severely penalizing malicious friends, SpamClean weights β by the square of the number of erroneous results discovery. In addition, SpamClean uses different penalty and recompense factors, and we propose to set $\beta > \alpha$. The selections of β and α are based on the specific requirement of application.

TABLE I: Experimental Configurations

Parameter	Meaning
PA	The probability that the normal user shares and annotates his resources.
PC	The total proportion of the collusive attackers
PD	The total proportion of the normal attackers.
PT	The total proportion of the tricky attackers.
FN	The number of the user's friends.

IV. Evaluation

In this section, we first describe the simulation setup, and then discuss the key performance metric. Finally, we evaluate the performance of SpamClean as compared to the existing tag search models: Boolean [6], Occurrence [7], and Coincidence [5].

A. Simulation Setup

We developed a prototype tagging system implementing all mechanisms of SpamClean. Furthermore, we need to generate several systems with different system parameters — all should follow certain distributions. Some important parameters throughout our simulations are described in TABLE I.

System Model: Because our focus is the quality of tag search results, in the following simulations, the transfer time is assumed to be negligible. Moreover, we also assume our prototype can work without the single point of failure.

Resource Model: In our simulation, there are $50,000$ resources, and the size of vocabulary, which is used by users to annotate their resources, is $30,000$. In order to generate a "full of spam" environment, each of the resources has only 10 correct tags in the vocabulary. At the startup of simulation, the selections of resources and annotations follow the distributions measured in [16] and [17] respectively.

User Model: The system is comprised of $5,000$ users including genuine users and malicious users (attackers). At the startup, the genuine user annotates the resources with the correct tags, and then participates in the system to search and collect the resources. The genuine users annotate all of their resources with the correct tags; whereas, the attackers share the erroneous annotations, and actively launch tag search in the system — this attempts to undermine the performance of tagging system. Throughout our simulation, genuine and malicious users may leave and rejoin the system.

Social Model: We generate the social network of simulation according to the small world property of online social networks [18], and establish the friend-relationships for the users based on widely adopted Kleinberg model [19]. In our experiments, we define the FN as the number of friends of one user in the system.

Execution Model: The different tag searches are initiated at uniformly distributed users in the tagging system. Our experiment is composed of 50 simulation cycles. In each cycle, the selection of $0-5$ specific tag is used to search by each user, and then the user selects and collects the resources according to the order of search results which are returned based on the mechanisms of SpamClean. After each simulation cycle, the number of spam search results is calculated. Withour the especially emphasized, we set $PA = 1$, $PC = 0$, $PD = 0.2$, $PT = 0$, and $FN = 6$ as the default configurations of our experiments. Each experimental simulation is run 5 times and the results of all runs are averaged.

B. Search Models

This section describes three respective tag search models: Boolean, Occurrence and Coincidence. In the following experiments, they are used to compare with SpamClean.

Boolean Model: Boolean is an easy search model which is used in some current tagging systems, e.g., Slideshare [6]. The search strategy of Boolean is that the system randomly ranks the results associated with the search tag.

Occurrence Model: Occurrence model (e.g., Rawsugar [7]) ranks the search results based on the number of annotations containing the searched tag, and returns the top ranking results.

Coincidence Model: Coincidence model [4], [5] assigns each user a global creditability score which is the sum of the same annotations between this user and the other users in the system, and then the system ranks the search results based on the average of all the annotators' creditability scores of each result.

C. Threat Models

In this section, we describe three threat models existing in the real-world tagging systems.

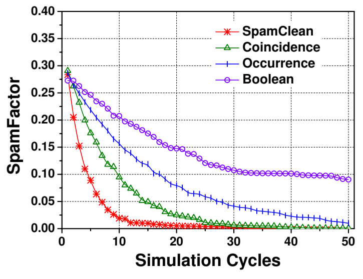
Normal Attack Model: For the random attackers, they always select some of the resources, and annotate these resources with the erroneous tags randomly to achieve the purpose of misleading the normal participants in the systems. Normally, the random attack acts independently, that is, these bad peers are "lousy taggers".

Collusive Attack Model: In some cases, the malicious users can collude and mount some intelligent attacks. These collusive attackers annotate the resources, which are collected by them in common, with the number of same misleading tags in order to make these resources easy to be searched by normal users.

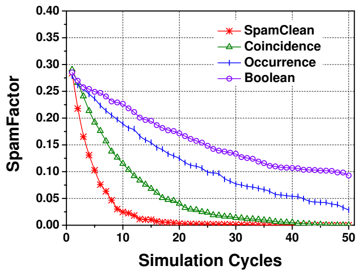
Tricky Attack Model: In the actual tagging systems, some malicious users may launch a tricky attack — they annotate the resources with both the correct and the erroneous tags, and then publish these annotations to the system. The existing anti-spam mechanisms (e.g., Coincidence) will be a victim when encountering this tricky attack.

D. Performance Metric

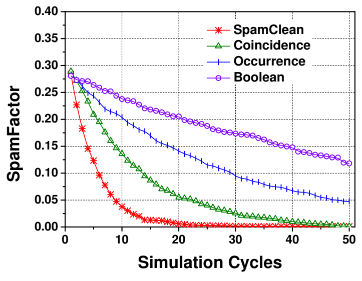
This section discusses the metric for evaluating our experiments. Because our purpose is to evaluate the impact of tag spam in the search result, we utilize *SpamFactor* [4], [5], a metric accepted widely to quantify the "spam impact" on search result. The study in [5] has argued that SpamFactor less than 0.1 is 'tolerable' in the sense that the spam resources will be few and towards the bottom of the result page. In our simulations, the SpamFactor focuses on the top20 search results.



(a) The Normal Attacks $PD = 0.1$

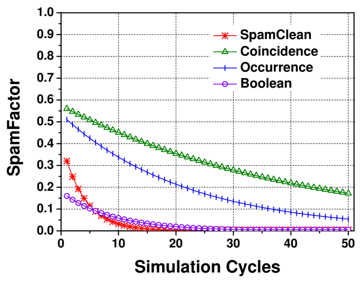


(b) The Normal Attacks $PD = 0.2$

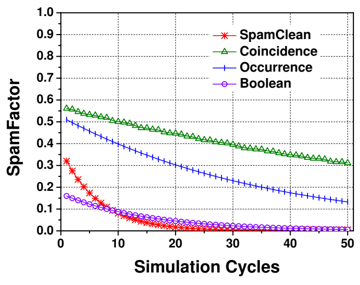


(c) The Normal Attacks $PD = 0.3$

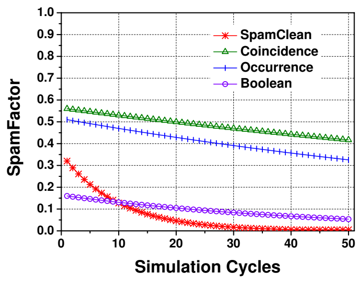
Fig. 1: Impact of the Normal Attacks.



(a) The Collusive Attacks $PC = 0.1$.



(b) The Collusive Attacks $PC = 0.2$.



(c) The Collusive Attacks $PC = 0.3$.

Fig. 2: Impact of the Collusive Attacks.

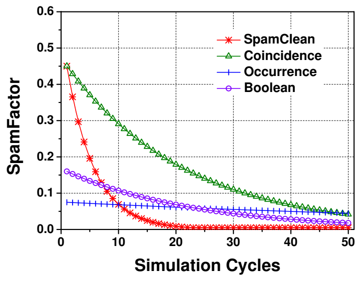
E. Evaluation Results

In this section, we first compare the SpamClean with the existing search models, Boolean, Occurrence, and Coincidence, both on the performance and the convergence under three attacks (mentioned in Section IV-C). Then, we evaluate SpamClean on the aspects of the correlation of users and impact of friend number respectively.

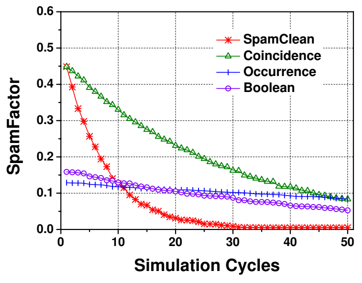
Impact of the Normal Attacks: In this experiment, we compare SpamClean with Boolean, Occurrence and Coincidence, both on the performance and the convergence under the environment with $PD = 0.1$, 0.2, and 0.3 respectively. First, we can notice that the result shown in Fig. 1 presents that the SpamFactor of Boolean is always higher than other models throughout the simulations. Meanwhile, the Occurrence is also influenced while increasing the proportion of the normal attackers, and especially Fig. 1c shows that, when setting PD to 0.3, the SpamFactor of Occurrence needs 30 simulation cycles to decrease below 0.1. On the other hand, both Coincidence and SpamClean can work well under the different values of PDs. Even when we set PD to 0.3, the SpamFactor of Coincidence can drop to 0.1 in 15 simulation cycles. Meanwhile, we can notice that SpamClean has the best performance and the convergence under the different values of PDs. Interestingly, the results in Fig. 1 shows that the performance of SpamClean remains almost unchanged under the different PDs. The reason is that, because that SpamClean client's search is ranked completely based on his own annotating behaviors, the SpamFactors of SpamClean clients' search results can not be influenced by the number of normal attackers in the system.

Impact of the Collusive Attacks: This experiment compares the performances of four models on defending against the collusive attackers in the tagging system, and, in the simulation, we set PC to 0.1, 0.2 and 0.3 respectively. The results shown in Fig. 2 present that the collusive attacks can make the serious influences on the both performances of the Coincidence and Occurrence. Throughout our simulations, the SpamFactors of Coincidence are always higher than 0.1 under three different values of PCs. Similar with the Coincidence, the SpamFactors of Occurrence also can not decrease to below 0.1, when we set PC to 0.2 and 0.3 respectively. Interestingly, the performances of Boolean have not been influenced by the collusive attacks. This is because that the mechanism of Boolean is to rank the search results randomly, the users choose the results attacked by the collusive attackers with the same probability as choosing the good results. At the startup, the performance of SpamClean is not very good; however, the SpamFactor of SpamClean can converge quickly to below 0.1 less than 10 simulation cycles, under all the three different values of PCs. From the experiments, we can observe that the good convergence of SpamClean is due to the socially-enhanced mechanism.

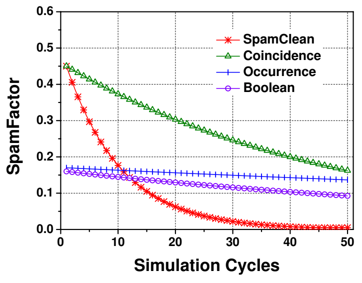
Impact of the Tricky Attacks: The result shown in Fig. 3 presents an interest phenomenon about both the performances of the Boolean and Occurrence. In Fig. 3a and Fig. 3b, we can notice that, at the startup, the Occurrence can work better than the Boolean; however, after 20 simulation cycles, the



(a) The Tricky Attacks $PT = 0.1$.



(b) The Tricky Attacks $PT = 0.2$.



(c) The Tricky Attacks $PT = 0.3$.

Fig. 3: Impact of the Tricky Attacks.

SpamFactor of Boolean is lower than Occurrence (both in Fig. 3a and Fig. 3b). More interestingly, Fig. 3c shows that, when we set PT to 0.3, the performance of Boolean is always better than the Occurrence throughout the simulation. The reason is that, as increasing of the proportion of tricky attackers, both the number of erroneous and correct annotations in the system also can be increased. Because the search of Occurrence is only based on the number of annotations, the SpamFactor of Occurrence always rises as increasing the number of erroneous annotations. On the other hand, due to the feature of Boolean searching (mentioned in Section IV-B), both the increasing number of erroneous and correct annotations can not influence the performance of Boolean specifically. Normally, the tricky attacks are launched specifically against the anti-spam mechanism; therefore, as shown in Fig. 3, Coincidence model can not provide the quality tag search results under the tricky attacks. We can also notice that, although the performances of SpamClean are influenced at the startup, the SpamFactors of SpamClean can converge quickly to below 0.1 in 8, 12, and 15 simulation cycles respectively.

Impact of the Cooperation of Normal Users: As our known, the tagging systems need the active cooperations among the participants. In this experiment, we evaluate the performance of SpamClean compared with Coincidence under the different probability that the normal users share and annotate their resources (PA). As shown in Fig. 4, when we set the PA to 1.0, the SpamFactors of SpamClean and Coincidence can decrease to below 0.1 in 5 and 12 simulation cycles respectively. However, while setting PA to 0.5, we can notice that both the performances of SpamClean and Coincidence are influenced by this low probability of the users' cooperations — the SpamFactors of two models are always higher than 0.1 before 30 simulation cycles. Therefore, we conclude that both SpamClean and Coincidence strongly depend on the cooperation of users' sharing and annotating, and SpamClean can performance with the better convergence due to the socially-enhanced mechanism.

Impact of the Number of Friends: Making use of the social network is the one of key mechanisms of SpamClean. In this experiment, we discuss the impact of each user's friend number (FN). As shown in Fig. 5, while setting the FN to 2, 4, 6, and 8 respectively, the performance of SpamClean

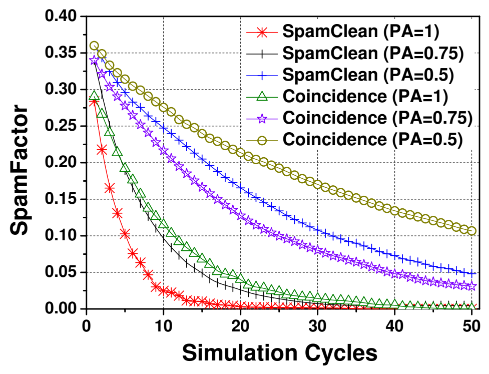


Fig. 4: Impact of the Cooperation of Normal Users

changes a lot. When each user only has two friends, the SpamFactor can decrease to below 0.1 after 15 simulation cycles; however, as our setting FN to 6, SpamClean can perform robustly to the good tag search results — the SpamFactor drops to below 0.1 in 6 simulation cycles only. Interestingly, when we vary the FN to 8, the enhancement of performance is not so prominent as the previous experiments (FN is set to 2, 4, and 6). This result indicates that each SpamClean client has 6 friends in the system is enough.

V. Conclusion and Discussion

Current tagging systems are highly vulnerable to tag spam. This paper proposes SpamClean, a novel social experience-based scheme towards spam-free and personalized tag search results in the tagging systems. Our evaluation results show that SpamClean can defend against tag spam from the various attackers more effectively than the existing search models — Boolean model, Occurrence model and Coincidence model. However, some other types of attacks can also be mounted against SpamClean in the real-word social networks, such as Sybil attack [8] and DoS attack.

Generally, the Sybil attack is an important security vulnerability in the current social networks — an attacker associates himself illegitimately with an arbitrary number of identities by impersonating other users or claiming false identities. Under Sybil attacks, the socially-enhanced mechanism of SpamClean

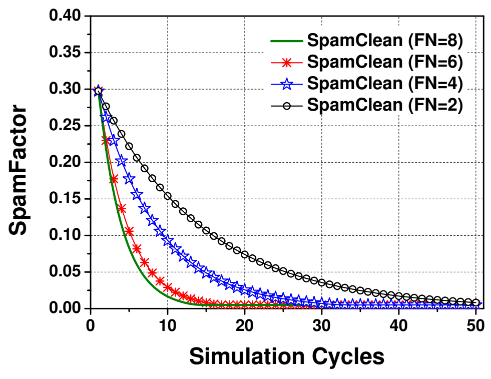


Fig. 5: Impact of Users' Friend Number

will be invalid. To protect against Sybil attacks, we could directly make use of the clients' social networks (trustworthy friend-relationships) to limit the number of Sybil attackers based on the mechanisms proposed in studies [20], [21]. As an alternative, we could also adopt the computational puzzle scheme to defend against the behaviors of generating tag spam of Sybil attackers[22], [23].

The DoS attack is another serious threat where one or more malicious users attempt to thwart normal participants from using the normal tagging services. The representative method launching a DoS attack is to cause the single point of failure. Specifically, in our work, the attackers can launch the flooding attacks to the centralized systems of tagging services, and the current SpamClean is easily to be a victim under this threat. The study in [24] indicated that the ingenious solution of current mechanisms against DoS attack is based on the calculation of puzzle scheme. Therefore, to defend against the DoS attack, SpamClean may require each client computes the moderate expense but not intractable puzzles to gain the admission to request tagging services frequently.

Bearing these in mind, we plan to incorporate SpamClean with the above security schemes to further improve her robust to the tag spam, and then extend her application field to some other categories of social networks, e.g., blog service.

Acknowledgment

We would like to thank Ruichuan Chen and Eng Keong Lua for helpful discussions. We would also like to thank the anonymous reviewers for their helpful suggestions. This work was supported in part by the NSFC under grant No. 60773163 and No. 60873238, as well as the NSFC & JSPS under grant No. 60911140102.

References

[1] "Flickr. http://www.flickr.com/."
[2] "Del.icio.us. http://del.icio.us/."
[3] "YouTube. http://www.youtube.com/."
[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*, vol. 11, no. 6, pp. 36–45, 2007.
[5] G. Koutrika, F. A. Effendi, Z. Gyöngyi, P. Heymann, and H. Garcia-Molina, "Combating Spam in Tagging Systems," in *AIRWeb*, 2007.
[6] "Slideshare. http://slideshare.net/."
[7] "Rawsugar. http://rawsugar.com/."
[8] J. R. Douceur, "The Sybil Attack," in *IPTPS*, 2002.
[9] B. Krause, C. Schmitz, A. Hotho, and G. Stumme, "The anti-social tagger: detecting spam in social bookmarking systems," in *AIRWeb*, 2008, pp. 61–68.
[10] Z. Kyriakopoulou and T. Kalamboukis, "Combining Clustering with Classification for Spam Detection in Social Bookmarking Systems," in *RSDC*, 2008.
[11] A. Gkanogiannis and T. Kalamboukis, "A novel supervised learning algorithm and its use for Spam Detection in Social Bookmarking Systems," in *RSDC*, 2008.
[12] T. Bogers and A. Bosch, "Using Language Models for Spam Detection in Social Bookmarking Systems," in *RSDC*, 2008.
[13] Z. Xu, Y. Fu, J. Mao, and D. Su, "Towards the semantic web: Collaborative tag suggestions," in *Collaborative Web Tagging Workshop in conjunction with WWW*, 2006.
[14] L. von Ahn, M. Blum, N. J. Hopper, and J. Langford, "CAPTCHA: Using Hard AI Problems for Security," in *EUROCRYPT*, 2003, pp. 294–311.
[15] O. Görlitz, S. Sizov, and S. Staab, "PINTS: Peer-to-Peer Infrastructure for Tagging Systems," in *IPTPS*, 2008.
[16] A. Mislove, M. Marcon, P. K. Gummadi, P. Druschel, and B. Bhattacharjee, "Measurement and analysis of online social networks," in *Internet Measurement Conference*, 2007, pp. 29–42.
[17] R. Li, S. Bao, Y. Yu, B. Fei, and Z. Su, "Towards effective browsing of large scale social annotations," in *WWW*, 2007, pp. 943–952.
[18] A. Mislove, M. Marcon, P. K. Gummadi, P. Druschel, and B. Bhattacharjee, "Measurement and analysis of online social networks," in *Internet Measurement Conference*, 2007.
[19] J. M. Kleinberg, "The small-world phenomenon: an algorithm perspective," in *STOC*, 2000.
[20] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman, "SybilGuard: defending against sybil attacks via social networks," in *SIGCOMM*, 2006, pp. 267–278.
[21] H. Yu, P. B. Gibbons, M. Kaminsky, and F. Xiao, "SybilLimit: A Near-Optimal Social Network Defense against Sybil Attacks," in *IEEE Symposium on Security and Privacy*, 2008, pp. 3–17.
[22] N. Borisov, "Computational Puzzles as Sybil Defenses," in *Peer-to-Peer Computing*, 2006, pp. 171–176.
[23] H. Rowaihy, W. Enck, P. McDaniel, and T. L. Porta, "Limiting Sybil Attacks in Structured P2P Networks," in *INFOCOM*, 2007, pp. 2596–2600.
[24] B. Parno, D. Wendlandt, E. Shi, A. Perrig, B. M. Maggs, and Y.-C. Hu, "Portcullis: protecting connection setup from denial-of-capability attacks," in *SIGCOMM*, 2007, pp. 289–300.