Addressing Malicious Noise in Clickthrough Data

Filip Radlinski
Department of Computer Science
Cornell University
Ithaca, NY, USA
filip@cs.cornell.edu

ABSTRACT

Clickthrough logs are becoming an increasingly used source of training data for learning ranking functions. Due to the large impact that the position in search results has on commercial websites, malicious noise is bound to appear in search engine click logs. We present preliminary work in addressing this form of noise, that we term click-spam. We analyze click-spam from a utility standpoint, and investigate the idea of whether personalizing web search results by partitioning the user population can reduce or eliminate the financial incentives for potential spammers. We formalize click-spam and analyze the incentives for malicious agents, then investigate the model with some examples.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Algorithms, Experimentation

Keywords

Click-Spam, Personalized Search, Clickthrough Data

1. INTRODUCTION

There has recently been an explosion of interest in using machine learning to improve web search (for example [4, 10, 13]). A particularly common approach is to use implicitly collected behavioral data to train such systems [6, 8]. One example is clickthrough data, which records queries issued by users to a search engine as well as the results clicked on by the users. Such implicitly collected data is particularly useful for improving the user experience because it comes from the users of the system, in contrast to web graph link structure (which comes from the much smaller group of authors) and in contrast to expert labeled data (which is expensive to obtain and may not reflect user intents). There have been a number of empirical studies that have shown that using clickthrough data is promising (for example [1, 10]).

Research to date for learning to rank has not addressed noise in the training data explicitly. Implicitly, all previous approaches that we are aware of have assumed random noise, which can safely be ignored when using modern machine learning techniques. However, for the past few years clickthrough data has been widely recorded by large commercial search engines, suggesting it may be used as part of the result ranking algorithm. As this clickthrough data starts to influence the results returned by these search engines, we believe that *malicious noise* will inevitably become present in clickthrough logs. It is well understood that since users tend to pay more attention to highly ranked sites, there is a strong financial incentive for a web site to be ranked highly in search results. Hence some website owners are very likely to attempt to create fraudulent clicks with the intention to modify the learned rankings in their favor. This is all the more likely as malicious noise is already pervasive in the link structure of the web graph [3], in online advertising [5] and in email communication. We refer to fraudulent clicks in clickthrough data as *click-spam*.

The standard technique to limit the impact of online spam today is to identify and remove it. However, it has become clear in the domain of Internet advertising that it is often difficult to identify fraudulent clicks on advertisements (although this is a very active area of research, see [9] for an overview). Similarly, many years of research into identifying spam emails has shown that as techniques for identifying spam improve, spammers find new techniques to avoid identification. In contrast, we propose to use search result personalization as a method to avoid the problem of click-spam altogether.

At a high level, the approach we investigate is to partition the user population into disjoint sets of users. Users in different partitions would see different search results. Such personalization has been shown to improve search performance and in fact be necessary for user satisfaction in web search to rise beyond a certain threshold [12] and has hence invited significant research. Our results here suggest that personalization can also be designed to limit or eliminate the financial incentives for click-spam. Clearly, a naive hope of partitioning users into spammers and non-spammers would attain such a goal, although finding such a partitioning is likely infeasible. Instead our models show that by finding even a weak partitioning, the financial incentive to create click-spam can be reduced or eliminated.

We start by describing a simplified model of web search that can be used to theoretically analyze the problem of click-spam. Next, we describe a formal utility based model of click-spam, discussing utility both from the users' and spammers' perspectives. Following this, we present conditions that determine whether click-spam is, or is not, in some malicious agent's interest. We continue with modeled results to analyze the practicality of partitioning a user population and finally conclude with a discussion of the implications of our findings and directions for future study.

2. MODELING SEARCH PREFERENCES

For many web search queries, we can model the ranking problem as follows. We have a large number of competitors C_i providing commercial services such as selling sporting goods, books or pharmaceuticals. Search engine users enter a query, and in response are presented with an ordered list of competitors. For such commercial searches, the user is much more likely to visit highly ranked sites than those ranked lower. For this reason, ensuring that the ranking is based on some reasonable measure (for example reputation, price or geographic proximity to the user) is important.

Since finding the best measures manually is difficult, a common approach is to use clickthrough data, which can be designed to provide a measure of the relevance of different competitors for the query. While there has been recent work into ensuring that clickthrough data represents relevance with minimal bias (e.g. [7, 11]), this work has assumed that the competitors being ranked do not attempt to exploit the system by generating fraudulent clickthrough data.

To make it possible to study such a system theoretically, consider a simpler setting where we have a single query and some set of m competitors C_1 through C_m. Let our algorithm for learning to rank convert clickthrough data into simple votes for each competitor C_i. This simplification is reasonable if we consider each vote as a training example and the aim is to maximize the fraction of votes satisfied.

A sensible approach for ranking the competitors would be to rank them in order of decreasing votes. To eliminate the obvious difficulties if one user is able to provide more votes than other users, we assume each user of the search engine can provide one vote. For example, if users are logged in to a search engine, the votes of each user can be normalized to sum to 1. If users to do not log in, the IP address or a cookie could be used as a proxy for user identity. We note that such normalization can be done conservatively, for example ignoring the votes coming through large proxies.

In this setting, a spammer can produce click-spam by taking other user's votes. In practice, this would involve fraudulent clicks caused by compromised systems or by paid users. We will call all users or systems creating malicious click data *spam hosts*. In particular, it could be in the interest of the lower ranked provider to produce click-spam if the cost of obtaining enough spam hosts to be ranked higher is less than the financial benefit of being ranked higher.

3. RANKING UTILITY

Search systems that learn from user preferences usually minimize the number of user preferences that are not satisfied (or an approximation or relaxation of this). We call the fraction of users whose preferences are satisfied by the ranking they are presented with the *user utility*. This is the measure that we are interested in optimizing in this work. However, there is also a utility of the ranking from the perspective of the competitors C_i. Each possible ranking has a specific value to each competitor being ranked. We call this utility the *competitor utility*. In this section, we formalize both the user utility and competitor utility.

3.1 User Utility

Assume that there are n users who are interested in buying the products that C_1 through C_m compete to sell. Let p^{C_i} be the percentage of users who prefer competitor C_i. Say we divide the population of users into k partitions, P_1 through P_k with $k << n$ and with each partition large enough that we do not need to consider small sample effects. The percentage of users in partition P_i is p_i, with $p_i^{C_j}$ being the percentage of all users who are in partition i and prefer C_j. For example if there are two competitors, and partition P_1 includes 10% of the users where three quarters of them prefer C_1 then $p_1 = 0.1$, $p_1^{C_1} = 0.075$ and $p_1^{C_2} = 0.025$.

We define the user utility of a partitioning as the fraction of users whose preferred competitor is ranked first if the learned ranking takes a majority vote in each partition. Without any click spam present, this is

$$util^u(P) = \sum_{i=1}^{k} \max_j p_i^{C_j} \tag{1}$$

In the case that click-spam is present, a spammer may flip which provider is ranked highest in any or all of the partitions. We define a spammer by η, the fraction of all hosts that the spammer turns into spam hosts. We assume the spammer can obtain any number of spam hosts, but restrict ourselves to spammers that do not a priori know which partition a particular host is in before it is compromised. This means that the additional number of votes that an η-spammer trying to promote C_s can gain is proportional to $\eta(1 - p^{C_s})$, since the remaining compromised hosts will already be voting for C_s. Within a particular partition P_i, an η-spammer could change the number of votes for C_s from $p_i^{C_s}$ to $p_{i,\eta}^{C_s} = p_i^{C_s} + \eta\left(p_i - p_i^{C_s}\right)$, gaining η of the votes in P_i not already cast for C_s. Substituting this modification for click-spam into the user utility, we can write the user utility given an η-spammer promoting C_s as

$$util^u(P) = \sum_i \begin{cases} p_i^{C_s} & \text{if } p_{i,\eta}^{C_s} > \max\limits_{j \neq s}(1-\eta) p_i^{C_j} \\ \max\limits_{j \neq s} p_i^{C_j} & \text{otherwise} \end{cases} \tag{2}$$

3.2 Competitor Utility

A primary consideration for most commercial websites is how many visitors their website attracts. We will use this as a measure of the utility of a ranking for each competitor. The number of visits can be related to where the competitor appears in the search result ranking, and will impact whether or not a competitor hires a spammer. Motivated by the approximately Zipfian form of the number of clicks on search results as a function of rank (for example see [2]), we define the utility of a ranking from C_i's perspective as the reciprocal rank of C_i averaged across all users. In effect, this counts the utility as a simple approximation of the number of clicks that C_i might receive. We define our pricing units such that the difference in utility for a competitor between being ranked first and second is 1 unit. Without any spam present, the competitor utility of a partitioning P to competitor C_s is

$$util^c(P, C_s) = 2n \sum_{i=1}^{k} p_i \left(1 + \left|\left\{j \ : \ p_i^{C_j} > p_i^{C_s}\right\}\right|\right)^{-1} \tag{3}$$

when there are n users in the population (the last term is the reciprocal rank of C_s in P_i). If C_s hires an η-spammer, the utility becomes

$$2n\sum_{i=1}^{k} p_i \left(1 + \left|\left\{j : (1-\eta)p_i^{C_j} > p_i^{C_s} + \eta(p_i - p_i^{C_s})\right\}\right|\right)^{-1} - \ cost(\eta - spammer) \tag{4}$$

To continue the analysis, we must assume a cost per spam host. Intuitively, compromising different hosts on the Internet has different costs. Some fraction of hosts can be compromised cheaply (e.g. old systems that have out of date software) while well maintained hosts behind a firewall would be very expensive to compromise. This suggests that the cost of obtaining spam-hosts should be superlinear in the number of hosts required. We model the cost of compromising a fraction η of all n hosts (equating each user to one host) with a simple quadratic function:

$$cost(\eta) = a\eta^2 n, \tag{5}$$

where a is a constant. Determining how the real cost of obtaining spam-hosts changes with the number of spam-hosts would be an interesting problem. To estimate a conservative value of a, say that 20% of hosts on the Internet can become spam hosts for an average cost of 1 per machine, i.e. the utility of being ranked at the top rather than second in a ranking for one user. In this case, $a = 25$.

4. THE ECONOMICS OF CLICK-SPAM

We can now ask the following questions: Given a partitioning P, when will it be profitable for a competitor to hire a spammer? What will be the impact on user utility? For the equations to be manageable, in this section we limit ourselves to the two competitor case, calling the first competitor A and the second B. We first derive a condition that must be satisfied for click-spam to be profitable, then we study the question of finding a good partitioning for a user population.

4.1 Theoretical Analysis

First consider the case when all users are in one partition $P = \{P_1\}$. $p_1^A = p^A$, $p_1^B = p^B$ and $p_1 = 1$. Further, without loss of generality assume that A has more votes than B, so with no spam, $util^A(P_1) = 2n$ and $util^B(P_1) = n$. If B hired a spammer, it would only make a difference if the spammer compromised $\frac{1}{2}(p^A - p^B)n$ hosts that vote for A and flipped the votes to be for B. The cost would be

$$cost(\eta) = a\left(\frac{1}{2}\frac{p^A - p^B}{p^A}\right)^2 n = an\left(1 - \frac{1}{2p^A}\right)^2 \tag{6}$$

Note the $p^A = 1 - p^B$ in the denominator comes from the spammer not knowing a host's vote before compromising it. Given that the value of a ranking where B has more votes is $2n$ (because B would now be ranked first for all users), it would be in B's interest to hire a spammer if and only if the additional value minus the cost is positive. Substituting Equation 6 into Equation 4 and solving for the one partition case (the extra reward is n, and cost is specified above), this happens when

$$p^A < \frac{1}{2}\frac{\sqrt{a}}{\sqrt{a} - 1} \tag{7}$$

The condition gets more complex when we have $k > 1$ partitions since hosts are compromised in *all* partitions when the spammer attacks *any* partition. Specifically, say we are given a partitioning P where in every partition either A or B has more votes. Let the partitions p_i be ordered such that $\forall i \in \{1, \ldots, k-1\}$, $p_i^A/p_i \leq p_{i+1}^A/p_{i+1}$. In words, the first partition has the largest fractional vote for B, followed by the second partition and so forth. This means that as η increases, an $\eta-$spammer would take over partitions in order, and guarantees that there is some t such that $i \leq t \Leftrightarrow p_i^B > p_i^A$.

Given this ordering of partitions, it will be in B's interest to hire a spammer if and only if the utility of taking over the j next partitions after P_t minus the cost of taking over the j^{th} partition is positive. The fraction η_j of machines that must be compromised to flip the outcome in partition P_j is simply half the margin by which A wins divided by the fraction of spam-hosts that were voting for A when taken over:

$$\eta_j = \frac{1}{2}\left(p_j^A/p_j - p_j^B/p_j\right)\frac{1}{p_j^A/p_j} = 1 - \frac{p_j}{2p_j^A}. \tag{8}$$

Thus B will hire a spammer if and only if

$$\exists j \ s.t. \ \left(\sum_{i=t+1}^{t+j} p_i\right) - a\left(1 - \frac{p_{t+j}}{2p_{t+j}^A}\right)^2 > 0 \tag{9}$$

The first term is the value to the spammer of taking over partitions $t+1$ through $t+j$ (note that A initially has more votes in each of them). The second term is the cost of taking over the $t + j^{th}$ partition. Since this partition has the largest imbalance, when it is taken over by the spammer, the spammer has also taken over all partitions with lower index.

4.2 Practical Partitioning

Above we presented a condition on partitions that tells us if spam is in the economic interest of competitor B in the case of two competitors and any number of partitions. We now turn to the question of whether partitioning would be useful in practice. We first present an example that demonstrates that partitioning a user population that would ordinarily be subject to click-spam can make spam no longer economically feasible. We follow that with a study of user utility for different partitionings.

4.2.1 *Illustrative Example*

Consider our example with two competing providers. Say 60% of users prefer competitor A and 40% prefer B. If we do not partition the population, 60% of the users would see their preferred provider first, meaning the user utility is 0.60. However, if we take $a = 25$, Equation 7 tells us that it would be in B's interest to hire a spammer (the right hand side is 0.625 while $p^A = 0.60$). Hence B would hire a spammer, who would reverse the ranking, reducing the user utility of the learned ranking to 0.40.

Say that we have a weak indicator that we can use to partition the users into three equally sized partitions ($p_1 = p_2 = p_3 = \frac{1}{3}$), with the fraction of users preferring A in each partition being 57%, 60% and 63% (i.e. $p_1^A = 0.19, p_2^A = 0.20, p_3^A = 0.21$). This partitioning has the same user utility as before, but it is no longer in B's interest to hire a spammer: To compromise enough hosts to tip the balance in

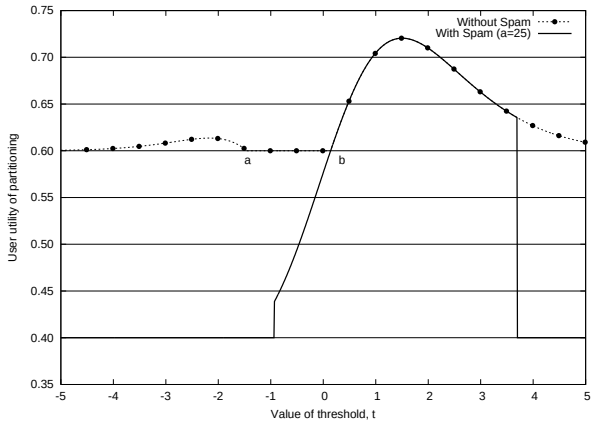

Figure 1: User utility as a function of threshold when $\mu_B = 1$ and $\sigma_B = 2$. $\mu_A = 0, \sigma_A = 1$ and $a = 25$.

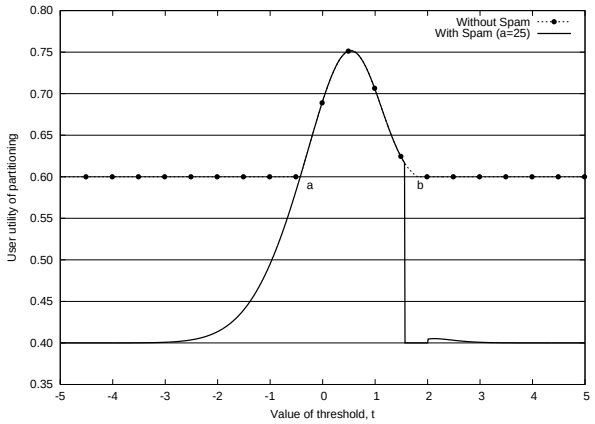

Figure 2: User utility as a function of threshold when $\mu_B = 1$ and $\sigma_B = 0.5$. $\mu_A = 0, \sigma_A = 1$ and $a = 25$.

P_1, the spammer would need to compromise approximately 12.3% of hosts (per Equation 8). This would cost approximately 0.377 while the payoff to B would only be 1/3. A similar calculation shows that it would not be in B's interest to tip the balance in the first two or all three partitions. Hence partitioning can increase robustness to spamming, even if the partitioning only changes the balance of votes in any partition by a small amount.

4.2.2 Constructing Partitions

We now extend the above simple illustration by presenting how such a partitioning could be found. Also, the above example partitioning does not result in an increased user utility relative to no partitioning when there is no spam. Partitioning is more interesting if it also increases the user utility. We study finding an optimal partitioning in the two-provider case, restricting ourselves to finding a partitioning of the user population into two groups for simplicity.

Assume that we can find a feature $f(u_i)$ for each user u_i that is weakly indicative of the preferences of the user. Say that users who prefer competitor A have $f(u_i)$ normally distributed with $f(u_i) \sim \mathcal{N}(\mu_A, \sigma_A^2)$, and similarly users who prefer competitor B have $f(u_i) \sim \mathcal{N}(\mu_B, \sigma_B^2)$. Then at any $x = f(u)$, the probability that a user prefers A is

$$p^A(x) = \frac{p^A \mathcal{N}(x; \mu_A, \sigma_A^2)}{p^A \mathcal{N}(x; \mu_A, \sigma_A^2) + p^B \mathcal{N}(x; \mu_B, \sigma_B^2)} \qquad (10)$$

where $\mathcal{N}(x; \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-(x-\mu)^2/2\sigma^2\right)$. Say we restrict ourselves to partitionings that pick a threshold t and assign all users with $f(u_i) \leq t$ to partition P_1 and all others to partition P_2. We would like to find an optimal threshold t that maximizes user utility even when there is click spam. In this preliminary work, we study the problem empirically. In the following, we take $p^A = 0.6$, $p^B = 0.4$ and pick a scale such that $\mu_A = 0$ and $\sigma_A = 1$. We also restrict ourselves to the case where any click-spam favors competitor B. We will study the effect of different values of μ_B and σ_B later.

Figure 1 shows the form of the user utility as the threshold t changes when the larger population (A) has $f(u)$ values with smaller variance than the smaller population (B). First consider the dotted line. This is the user utility as function of threshold t when click-spam is not present. Note that without partitioning, the user utility would be 0.6, since 60% of the population belong to the majority class (A). By partitioning, we can increase the user utility. Since $\sigma_B > \sigma_A$, B has a heavier tail than A. Hence there exists a threshold b above which a higher fraction of users prefer B than A. This point is marked on the figure. If $t > b$, the ranking shown to users in partition P_2 ranks competitor B above A. The form of the plot comes about since as t gets large, the absolute number of users in P_2 becomes small so the gain in user utility becomes small. On the other hand, if t is near b, there are still many users who prefer A in P_2 and this reduces the overall user utility. We see a similar effect at $t < a$. However, users who prefer B tend to be near $f(u) = \mu_B = 1$, so the number of users in the tail below the point where the population preferring B outnumbers that preferring A is small, leading to a smaller increase in user utility.

The solid line in Figure 1 shows the effect of click-spam, when there is a spammer who is hired by B. Note that in the unpartitioned case, the cost of spam is such that it is in B's interest to hire the spammer, thus reducing the overall user utility to 0.4 (since 40% of the user population actually prefers B). We see that with the partitioning, the choice of threshold has a dramatic impact on the user utility when click-spam is present. At the extremes, when the threshold is either very small or very large, we essentially have one small partition dominated by users who prefer B and one large partition dominated by users who prefer A. When the large partition is large enough, despite the fraction of users who prefer A being somewhat increased, click-spam is still economical per Equation 7. In the third regime, when $a \leq t \leq b$, the partitioning creates two partitions where A dominates in both but by different margins. For some thresholds it is only in B's interest to dominate just one of the partitions, resulting in a user utility between 0.4 and 0.6, while at others B dominates both.

Figure 2 shows a similar plot except with $\sigma_B < \sigma_A$. This means that the users in the smaller population (who prefer B) are well clustered. As a result, for large and small t, users who prefer A dominate both partitions. For intermediate threshold values, when most of the users who prefer B are in the same partition while a significant number of users who prefer A are eliminated from that partition, we see an

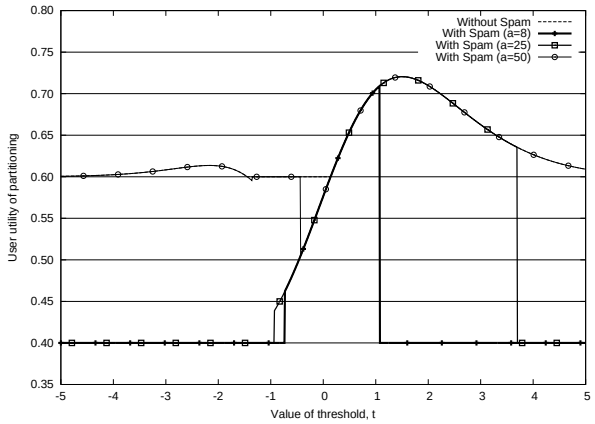

Figure 3: Effect of different costs for click-spam on the user utility as a function of threshold between partitions. Plot generated with $\mu_A = 0, \sigma_A = 1, \mu_B = 1, \sigma_B = 2$

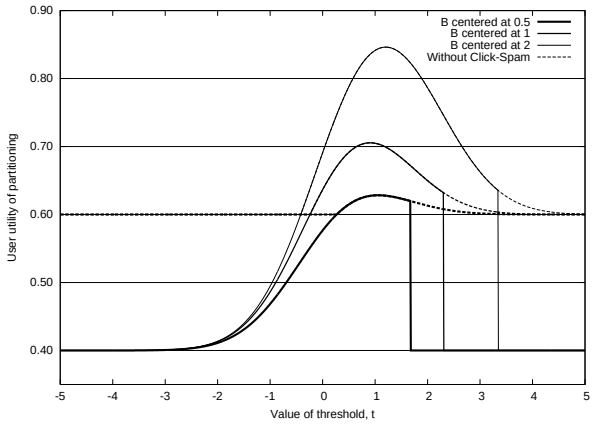

Figure 4: Effect of partition separation on user utility. Plot generated with $\mu_A = 0, \sigma_A = 1, \sigma_B = 1$ and $a = 25$

improvement in user utility. However, when the threshold is outside the critical range, the spammer is always able to attack at least one of the partitions (users who prefer A dominate both). Within the range, we see that for many thresholds the spammer cannot influence the user utility.

To summarize, we see that partitioning by thresholding on a weak feature can improve user utility and also can make the ranking function less prone to click-spam. However, we also see that in selecting the threshold, click-spam must be taken into account. Note that the case not shown in these figures, when $\sigma_B = \sigma_A$, is shown in Figure 4.

In Figures 1 and 2, the maximum user utility in the non-spam case has corresponded to a threshold where it is not in B's interest to spam. Unfortunately this is not always the case. To find the optimal threshold for partitioning, we need to find the threshold that has the highest user utility given that there may be click-spam. Figure 3 shows the effect of the cost a on the user utility in the presence of click-spam. Shown here are curves for the user utility with spam for three values of a. We see that in general, thresholds that are most prone to clickspam are those that correspond to lower user utility without click-spam. However, if the cost of clickspam is sufficiently low (as seen in the $a = 8$ curve), optimizing for user utility without taking into account clickspam may result in a partitioning that can be spammed. In particular note that with $a = 8$, at the threshold that maximizes user utility without click-spam, with click-spam the user utility drops to the minimum. The threshold value of a for which this effect happens rises as the difference between overall votes for A and B diminishes.

Finally, we briefly consider how the informativeness of the weak feature affects the choice of optimal threshold. We see in Figure 4 that as the separation between the two user populations changes, the optimal threshold changes. In particular, we see that as the separation between the two populations decreases, click-spam affects the optimal more and the maximal overall improvement in user utility from partitioning decreases.

5. CONCLUSIONS AND FUTURE WORK

In this paper we have described the problem of clickspam that we believe is likely to appear in future clickthrough data commonly used for learning to rank. We have presented a model for analyzing the utility of clickspam and determining if clickspam will indeed be a problem. Our results show that personalizing search results, which we model by partitioning the user population, can improve the robustness to clickspam when using a simple ranking algorithm. We postulate that personalized search is a good approach for avoiding the problem of clickspam in addition to its use for increasing user satisfaction in terms of search results.

We plan to extend the current empirical analysis to provide theoretical results on optimal partitioning for this click-spam model. In particular, it would be interesting to find an optimal set of thresholds $\{t_i\}$ that should be used to partition a user population with k classes of users given a weak indicator of class membership when clickspam is present. Additionally, throughout the empirical analysis, we only considered a spammer hired by one competitor. We plan to extend this work to consider all competitors as potential spammers. The complete optimization problem will involve all the competitors making a decision whether or not to hire a spammer with the potential for complex interactions (for example, the effectiveness of one competitor spamming will change if another competitor also spams and changes the vote balance).

6. ACKNOWLEDGMENTS

The first author was supported by a Microsoft Research Fellowship. This work was also partially supported by NSF Career Award 0237381 and a gift from Google. The author thanks Thorsten Joachims for the initial idea to consider click-spam from a utility standpoint and further discussions and feedback, and the anonymous reviewers for detailed and helpful feedback.

7. REFERENCES

[1] E. Agichtein, E. Brill, and S. Dumais. Improving web search ranking by incorporating user behavior information. In *Proceedings of the ACM Conference*

on Research and Development in Information Retrieval (SIGIR), 2006.

[2] E. Agichtein, E. Brill, S. Dumais, and R. Ragno. Learning user interaction models for predicting web search result preferences. In *Proceedings of the ACM Conference on Research and Development in Information Retrieval (SIGIR)*, 2006.

[3] L. Becchetti, C. Castillo, D. Donato, and S. Leonardi. Link-based characterization and detection of web spam. In *Adversarial Information Retrieval on the Web Workshop at SIGIR 2006*, 2006.

[4] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2005.

[5] B. J. Jansen. Adversarial information retrieval aspects of sponsored search. In *Adversarial Information Retrieval on the Web Workshop at SIGIR 2006*, 2006.

[6] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining (KDD)*, 2002.

[7] T. Joachims, L. Granka, B. Pan, H. Hembrooke, F. Radlinski, and G. Gay. Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *ACM Transactions on Information Science (TOIS)*, 25(2), April 2007.

[8] D. Kelly and J. Teevan. Implicit feedback for inferring user preference: A bibliography. *SIGIR Forum*, 32(2), 2003.

[9] A. Metwally, D. Agrawal, and A. E. Abbadi. Detectives: Detecting coalition hit inflation attacks in advertising networks streams. In *Proceedings of the International World Wide Web Conference (WWW)*, pages 241–250, 2007.

[10] F. Radlinski and T. Joachims. Query chains: Learning to rank from implicit feedback. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD)*, 2005.

[11] F. Radlinski and T. Joachims. Minimally invasive randomization for collecting unbiased preferences from clickthrough logs. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2006.

[12] J. Teevan, S. T. Dumais, and E. Horvitz. Beyond the commons: Inverstigating the value of personalizing web search. In *Workshop on New Technologies for Personalized Information Access (PIA 2005)*, 2005.

[13] Z. Zheng, H. Zha, K. Chen, and G. Sun. A regression framework for learning ranking functions using relative relevance judgments. In *Proceedings of the ACM Conference on Research and Development in Information Retrieval (SIGIR)*, 2007.