# Blocking Blog Spam with Language Model Disagreement

Gilad Mishne
Informatics Institute, University of Amsterdam
Kruislaan 403, 1098SJ Amsterdam
The Netherlands
gilad@science.uva.nl

David Carmel, Ronny Lempel
IBM Research Lab in Haifa
Haifa 31905, Israel
{carmel,rlempel}@il.ibm.com

## ABSTRACT

We present an approach for detecting link spam common in blog comments by comparing the language models used in the blog post, the comment, and pages linked by the comments. In contrast to other link spam filtering approaches, our method requires no training, no hard-coded rule sets, and no knowledge of complete-web connectivity. Preliminary experiments with identification of typical blog spam show promising results.

## Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval - search engine spam; I.7.5 [**Document Capture**]: Document analysis - document classification, spam filtering; K.4.1 [**Computers and Society**]: Public Policy Issues - abuse and crime involving computers, privacy

## General Terms

Algorithms, Languages, Legal Aspects

## Keywords

Comment spam, language models, blogs

## 1. INTRODUCTION

The growing popularity of internet search as a primary access point to the web has increased the benefits of achieving top rankings from popular search engines, especially for commercially-oriented web pages. Combined with the success of link analysis methods such as PageRank, this led to a rapid growth in link spamming – creating links which are "present for reasons other than merit" [6]. A well-known example of link spamming is link farms – link exchange programs existing for the sole purpose of boosting the link-based prestige of participating sites; these farms are fairly easily identified by topological analysis. However, in recent years search engines are facing a new link spam problem which is harder to track down: *comment spam*.

Comment spam is essentially link spam originating from comments and responses added to web pages which support dynamic user editing. With the massive increase in the number of blogs in recent years, such pages have proliferated; additional editable web pages which are often the target of comment spammers are wikis [1] and guestbooks. Blogs have made the life of comment spammers easy: instead of setting up complex webs of pages linking to the spam page, the spammer writes a simple agent that visits random blog and wiki pages, posting comments that link back to her page. Not only is spamming easier, but the spammer also benefits from the relatively high prestige that many blogs enjoy, stemming both from the rapid content change in them and the density of links between them. Comment spam, and link spam in general, poses a major challenge to search engines as it severely threatens the quality of their ranking. Commercial engines are seeking new solutions to this problem [9]; accordingly, the amount of research concerning link spam is increasing [8, 2, 6].

In this paper, we follow a language modeling approach for detecting link spam in blogs and similar pages. Our intuition is simple: we examine the use of language in the blog post, a related comment, and the page linked from the comment. In the case of comment spam, these language models are likely to be substantially different: the spammer is usually trying to create links between sites that have no semantic relation, e.g., a personal blog and an adult site. We exploit this divergence in the language models to effectively classify comments as spam or non-spam. Our method can be deployed in two modes: in retrospective manner, when inspecting a blog page which already has comments (this is particularly useful for crawlers); or in an online manner, to be used by the blog software to block spammers on the fly.

The rest of the paper is organized as follows. In Section 2 we survey existing work in the area of link spam. Our language modeling approach is presented and formalized in Section 3. Section 4 follows with a description of preliminary experiments conducted using this method; we conclude in Section 5. This paper discusses ongoing work and is meant to provide the conceptual framework and initial results, rather than a complete analysis of our proposed solution.

## 2. RELATED WORK

### 2.1 Comment Spam

Most approaches to preventing comment spam are technical in nature, and include:

- Requiring registration from comment posters;

[1]Collaborative websites whose content can be edited such as Wikipedia – http://wikipedia.org


*AIRWeb'05*, May 10, 2005, Chiba, Japan.

- Requiring commenters to solve a *captcha* – a simple Turing test mechanism [17];
- Preventing HTML in comments;
- Preventing comments on old blog posts;
- Using lists of forbidden or allowed IP addresses for commenters ("blacklists" and "whitelists");
- Using internal redirects instead of external links in comments;
- Limiting the number (or rate) of comments being added ("throttling").

While some of these mechanisms can be quite effective, they also have disadvantages. Methods which require user effort such as registration reduce the number of spontaneous responses which are important to many blog maintainers. Additionally, they do not affect the millions of commented web pages already "out there", and only address new comments. Preventing commenting altogether, or limiting it to plain text, or enforcing redirects on links in it, limits also legitimate comments and links contained in them, reducing the effectiveness of link analysis methods. Blacklists and whitelists require constant maintenance, and are bypassed by spammers using proxies and spoofed legitimate IP addresses. Throttling can reduce the amount of spam from a single page, but not the phenomenon altogether; spammers will simply post to more blogs.

Recently, a number of major search engines such as Yahoo, MSN Search and Google announced that they are collaborating with blogging software vendors and hosts to fight comment spam using a special attribute added to hypertext links [13]. This tag, `rel="nofollow"`, tells search engines that the links are untrusted, and will effectively prevent application of link-analysis scores to links associated with it – maybe even prevent crawling them altogether. However, the usage of this attribute is problematic for a number of reasons, including harming the inter-linked nature of blogs and possible abuse by webmasters; indeed, it is disputed within the blogging community [14, 4], and many do not intend to use it (e.g., at the time of writing, Yahoo's own search blog – which announced the tag – does not implement it).

## 2.2 Content Filtering and Spam

A different set of approaches for fighting comment spam works by analyzing the content of the spam comment, and possibly also the contents of pages linked by the comment (e.g., [12]). All these techniques are currently based on detecting a set of keywords or regular expressions within the comments. This approach suffers from the usual drawbacks associated with a manual set of rules, i.e. high maintenance load as spammers are getting more sophisticated. Typically, content-based methods require training with a large amount of spam and non-spam text, and correcting mistakes that are made; failure to continuously maintain the learner will decrease its accuracy, as it will create an inaccurate conception of what's spam and what's not. Having said that, regular expression based methods are fairly successful currently, partly due to the relatively young history of link spamming. It is expected that, similarly to the email spam world, as comment spammers enhance their methods, rule-based approaches will become less effective.

Published work on spam refers mostly to email spam, which was popular long before comment spam was, and was therefore targeted from many industrial and academic angles. In the email domain, machine learning and language modeling approaches have been very effective in classifying spam [10, 3]. An important difference between email spam and comment spam stems from the fact that comment spam is *not intended for humans*. No comment spammer actually expects anyone to click on the link that was added: this link is meant solely for the purpose of being followed by web crawlers. Thus, the spammer can (and does) use any type of words/features in his comment: the main goal is to have the link taken into account by search engine ranking schemes, and strings which have been reported as good discriminators of email spam such as over-emphasized punctuation [16] are not necessarily typical of comment spam.

## 2.3 Identifying Spam Sites

An altogether different approach to spam filtering is not to classify individual links as spam links or legitimate links, but to classify pages or sites as spam; recent work in this area includes usage of various non-content features [8] and link analysis methods [2]. The drawback of these approaches is that spam is essentially not a feature of pages, but of links between pages; sites can have both legitimate and spam incoming links (this is true for many online shops). Additionally, usage of some of these methods requires full connectivity knowledge of the domain, which is beyond the abilities of most bloggers.

In comparison to the existing methods presented, our approach requires no training, no maintenance, and no knowledge of additional information except that present on the commented web page.

# 3. COMMENT SPAM AND LANGUAGE MODELS

In this section we outline our language model based approach to identifying comment spam.

In the previous section, we noted that email spam is easier to classify than comment spam since it tends to have characterizing features – features which are supposed to convince a human to respond to the spam mail. On the other hand, comment spam has an advantage (from the filtering perspective) that email spam does not have. While every email needs to be classified as spam in an isolated manner, blog comments are presented within a *context*: a concrete semantic model in which the comment was posted. Our main assumption is that spam comments are more likely to violate this context by presenting completely different issues and topics. We instantiate the semantic models of the context, the comment and the page linked by the comment using language models.

## 3.1 Language Models for Text Comparison

A language model is a statistical model for text generation: a probability distribution over strings, indicating the likelihood of observing these strings in a language. Usually, the real model of a language is unknown, and is estimated using a sample of text representative of that language. Different texts can then be compared by estimating models for each of them, and comparing the models using well-known methods for comparing probability distributions. Indeed, the use of language models to compare texts in the Information Retrieval setting is empirically successful and becoming

increasingly popular [15, 11].

As noted earlier, we identify three types of languages, or language models, involved in the comment spam problem (see Figure 1). First, there is the model of the original blog post. Then, every comment added to the post adds two more models: the language used in the comment, and the language used in the page linked by the comment.

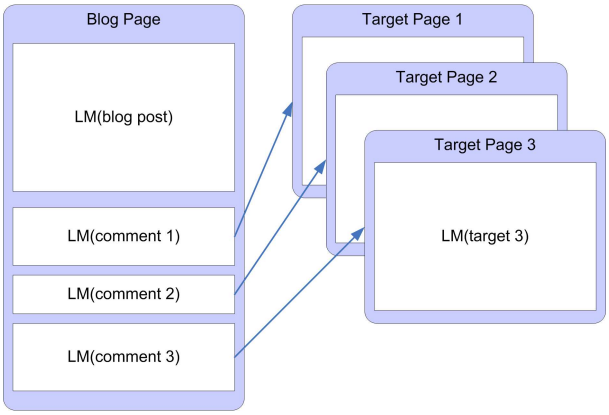


**Figure 1: Three types of language models: the model of the blog post, the models of the comments, and the models of the pages linked by the comments.**

To compare the models, we follow a variation on the Interpolated Aggregate Smoothing used by Allan *et. al.* in [1]. In practice, this measure calculates the smoothed Kullback-Leibler divergence between the language model of a short fragment of text and a combined language model of knowledge preceding this text. Formally, the KL-divergence between two probability distributions $\Theta_1, \Theta_2$ is

$$KL(\Theta_1 \| \Theta_2) = \sum_w p(w|\Theta_1) \log \frac{p(w|\Theta_1)}{p(w|\Theta_2)}$$

where $p(w|\Theta_i)$ is the probability of observing the word $w$ according to the model $\Theta_i$. In Interpolated Aggregate Smoothing, probabilities are estimated using maximum likelihood models and smoothed with Jelinek-Mercer smoothing. The two language models we are comparing are any pair of the triplet (blog post, comment, page linked by commet); let's examine the comparison between the model of the blog post ($\Theta_P$), and the model of a comment to this post ($\Theta_C$). We estimate the probabilities using maximum likelihood and smooth using a general probability model of words on the internet obtained from [7]. This gives us:

$$\begin{aligned} p(w|\Theta_P) &= \lambda_1 p(w|\Theta_{ML(post)}) \\ &+ (1-\lambda_1) p(w|\Theta_{ML(internet)}) \end{aligned}$$

$$\begin{aligned} p(w|\Theta_C) &= \lambda_2 p(w|\Theta_{ML(comment)}) \\ &+ (1-\lambda_2) p(w|\Theta_{ML(internet)}) \end{aligned}$$

Where $ML(\langle source \rangle)$ are maximum likelihood estimates. The language models used here are unigram models, but it is possible to use n-grams of higher orders in the same manner.

## 3.2 Spam Classification

Once we have a language model for each comment and a score based on its similarity to the language model of the blog post, we use these scores to classify the comment as spam or non-spam. Although this can be done using a simple threshold, we follow a different approach. We assume that the spammed blog page is parsed by a crawler, and the crawler is trying to assess which links (from the comments) should be used for link analysis methods and which not. In this case, the crawler views a range of comments, and must distinguish the good ones from the bad. As a first stage, the crawler calculates the KL-divergence for all comments as indicated above. The values obtained can then be seen as drawn from a probability distribution which is a mixture of Gaussians: each Gaussian represents a different language model. The Gaussian with the lowest mean – the least distance from the language model of the original blog post – represents the language model which is closest to the original post. Subsequent Gaussians represent language models which have a larger deviation from the original one, and are therefore more likely to constitute spam comments.

For our model, we assume the KL-divergence scores to be drawn from a 2-Gaussian distribution: the "legitimate" language model, and all other (spam) models (see example of the distribution in one of the blog pages in Figure 2). To estimate the parameters of the Gaussians, we use the EM algorithm.

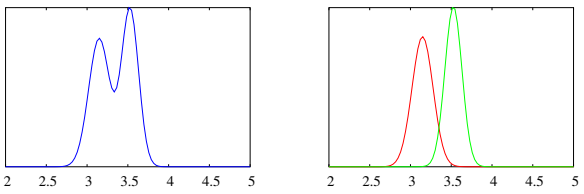


**Figure 2: Gaussian mixture model estimated from the KL-divergence values of 10 comments on a blog post, and its underlying Gaussians**

Finally, a comment is classified as spam if its KL-divergence from the blog post is more likely to be drawn from the spam Gaussian than from the legitimate one. For this purpose, we calculate a discriminating value between the two Gaussians – a number for which lower values are more likely to be drawn from the Gaussian with lower mean, and higher values are more likely to be drawn from the other Gaussian. Visually, this threshold can be viewed as the best vertical separator between the two distributions. Note that this threshold value provides us with an easy mechanism for changing the likelihood of identifying false positives ("legitimate" comments classified as spam) and false negatives (unidentified spam). Decreasing the threshold ("moving" the separator left) will result in a more strict requirement from the language model divergence between the comment and the post, effectively increasing the number of false positives and reducing false negatives; increasing the threshold value ("moving" the line to the right) will cause our method to be more tolerant to higher KL-divergence values, reducing false positives at the cost of increased false negatives. Usually, the cost of false positives is considered higher than that of false negatives; in general, we can use a *threshold multiplier* value to adjust the original threshold (where a multiplier of 1.0 will leave the threshold unchanged).

## 3.3 Model Expansion

Blog comments can be very short, and this is true also for some blog posts. This results in sparse language models, containing a relatively low number of words. We therefore propose to enrich the models of both the post and the comment, to achieve a more accurate estimation of the language model. An intuitive way to do so is to follow links present in the post and the comment, and add their content to the post and the comment, respectively; in the case of the post, it is also possible to follow incoming links to the blog and add their content. Taking this a step further, it is also possible to continue following links up to depth N, although this potentially causes topic (and language model) drift.

## 3.4 Limitations and Solutions

An easy way for spammers to "cheat" our model (or any other model which compares the contents of the post and the comment) is to generate comments with a similar language model to the original blog post. This makes the link-spam bots slightly more complicated since they must identify the post contents and use its model for generating a close one (e.g., by copying phrases) – but spammers have shown to overcome much higher obstacles.

However, in this case of language model faking, a new opportunity arises: assuming the spammer posts multiple comments in many different blogs (as a means of increasing the PageRank), there are now many comments with completely different language models to the same spam site. This is easily detectable at the search engine level which has a connectivity server at its hand; it can also be detected by an iterative HITS-like method by the blogger, following the link to the spam site and then its incoming links.

As any other spam filtering method, ours is not foolproof and can make mistakes; comments which are formulated with a sufficiently different vocabulary than the blog post might be mistaken for spam. However, this is precisely the reason for the robustness of our approach: it is very hard for the spammer to create comments that will be both similar to the blog language and to the spam site language. To account for these mistakes, an alternative to using our method as a binary spam/non-spam classifier is to use it for assigning a weight to links found in comments according to their language model divergence; the weight can be used to decrease PageRank of malicious sites, using methods such as the one reported in [5].

# 4. EXPERIMENTAL SETTING

We now present some preliminary experiments in identifying comment spam in blogs using our approach.

We collected 50 random blog posts, along with the 1024 comments posted to them; all pages contain a mix of spam and non-spam comments. The number of comments per post ranged between 3 and 96, with the median being 13.5 (duplicate and near-duplicate comments were removed). We manually classified the comments: 332 (32%) were found to be "legitimate" comments, some of them containing links to related pages and some containing no links; the other 692 comments (68%) were link-spam comments [2].

The link-spam comments we encountered in our corpus are of diverse types; while some of them are simple keyword lists, accompanied by links to the spam sites, others employ more sophisticated language (see Figure 3 for some sample comments). A typical blog page from our corpus contains a mix of different comment spam types.


**Figure 3: Samples of comment spam in our collection (top to bottom): [1] keyword based, with a random number to prevent duplicate detection; [2] revealing internal implementation of the spamming agent; [3] using quotes – in this case, from Douglas Adams – as "trusted" language; [4] disguising as random surfer**

In this set of experiments, we compared only the language models of the post and the comments, and did not take into account the model of the page linked by the comments. This was done due to time constraints. The values of $\lambda_i$ (see previous section) were both set to 0.9.

## 4.1 Results

The results of our experiments are shown in Table 1. False negatives are spam comments which were not identified as spam by our method, while false positives are non-spam comments which were classified as spam. The threshold multiplier is the value used to modify the separator between the two language models as described in section 3.2.

As a naive baseline, we use the maximum likelihood probabilities for the comment type in our model; as noted earlier, 68% of the comments were spam, so we assume an ad-hoc fixed probability of 0.68 for a comment to contain link spam. We achieve reasonable performance with our model, and can clearly see the trade-off between misclassifying spam and misclassifying non-spam, resulting from different modifications to the language model threshold.

*Discussion.* The size of our corpus is far from being satisfactory: we therefore label our results as a proof-of-concept and basis for continued experimentation, rather than full-fledged evidence of the method's capabilities. Nevertheless, the results are encouraging and clearly show that our in-

[2]The collection can be obtained from http://ilps.science.uva.nl/Resources/blogspam/

Table 1: Blog link-spam classification results

| Method | Threshold Multiplier | Correct | False Negatives | False Positives |
|---|---|---|---|---|
| Baseline (avg. 100 runs) | N/A | 581 (57%) | 223 (21.5%) | 220 (21.5%) |
| KL-divergence | 0.75 | 840 (82%) | 65 (6.5%) | 119 (11.5%) |
| KL-divergence | 0.90 | 834 (81.5%) | 69 (6.5%) | 121 (12%) |
| KL-divergence | 1.00 | 823 (80.5%) | 88 (8.5%) | 113 (11%) |
| KL-divergence | 1.10 | 850 (83%) | 87 (8.5%) | 87 (8.5%) |
| KL-divergence | 1.25 | 835 (81.5%) | 111 (11%) | 78 (7.5%) |

tuition is correct: the language used in spam comments does diverge from the language of the blog post substantially more than the language used in legitimate comments.

An analysis of the misclassified comments reveals that many of them are very short – containing 3-4 words, usually a non-content response to the post (e.g., "That sounds cool"). However, the vast majority of these comments contain no external links, or an email link only – so their misclassification will not result in actual search engine spam (in the case of false negatives) and not change the "true" link-analysis prestige of pages (in the case of false positives). While it is possible to integrate language divergence with comment length and other features into a hybrid comment spam classification system, we focused on the language aspect only and did not explore usage of additional knowledge.

*Model Expansions.* As mentioned earlier, a possible solution to the sparseness of some of the blog posts is to expand the language model in various ways. We performed a limited amount of experiments involving such expansions, by following all links present in the blog post and adding the content present in the target pages to the content of the blog post, before estimating the language model. Of the 50 blog posts in our corpus, 31 posts had valid links to other pages (some posts did not contain links at all, and some contained expired and broken links). The average number of links followed (for the 31 pages with expansions) was 3.4. Unfortunately, using the expanded models did not improve the overall classification accuracy. In fact, while for some blog posts – most notably shorter ones – the expansion helped substantially, we experienced a degradation of 2%-5% in the average performance over the entire corpus. However, both the fairly small number of pages which were expanded and the limited experiments performed prevent us from formulating a definite statement regarding model expansion at this stage.

## 5. CONCLUSIONS

We presented an approach for classifying blog comment spam by exploiting the difference between the language used in a blog post and the language used in the comments to that post (and pages linked from those comments). Our method works by estimating language models for each of these components, and comparing these models using well-known methods. Preliminary experiments using our method to classify typical comment spam show promising results; while in this paper we discuss blogs, the problem and the solution are relevant to other types of comment spam, such as wiki spam.

## 6. REFERENCES


[1] J. Allan, C. Wade, and A. Bolivar. Retrieval and novelty detection at the sentence level. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 314–321. ACM Press, 2003.

[2] E. Amitay, D. Carmel, A. Darlow, R. Lempel, and A. Soffer. The connectivity sonar: detecting site functionality by structural patterns. In *HYPERTEXT '03: Proceedings of the fourteenth ACM conference on Hypertext and hypermedia*, pages 38–47. ACM Press, 2003.

[3] I. Androutsopoulos, J. Koutsias, K. V. Chandrinos, and C. D. Spyropoulos. An experimental comparison of naive bayesian and keyword-based anti-spam filtering with personal e-mail messages. In *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 160–167. ACM Press, 2000.

[4] Nofollow tag cheers bloggers, but fails blogs, URL: `http://www.platinax.co.uk/news/archives/2005/01/new_nofollow_ta.html`.

[5] R. Baeza-Yates and E. Davis. Web page ranking using link attributes. In *WWW Alt. '04: Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, pages 328–329. ACM Press, 2004.

[6] B. Davison. Recognizing nepotistic links on the web. In *AAAI-2000 workshop on Artificial Intelligence for Web Search*, pages 23–28. AAAI Press, 2000.

[7] The Berkeley/Stanford Web Term Document Frequency and Rank project, URL: `http://elib.cs.berkeley.edu/docfreq/`.

[8] D. Fetterly, M. Manasse, and M. Najork. Spam, damn spam, and statistics: using statistical analysis to locate spam web pages. In *WebDB '04: Proceedings of the 7th International Workshop on the Web and Databases*, pages 1–6. ACM Press, 2004.

[9] M. R. Henzinger, R. Motwani, and C. Silverstein. Challenges in web search engines. *SIGIR Forum*, 36(2):11–22, 2002.

[10] J. M. G. Hidalgo. Evaluating cost-sensitive unsolicited bulk email categorization. In *SAC '02: Proceedings of the 2002 ACM symposium on Applied computing*, pages 615–620. ACM Press, 2002.

[11] V. Lavrenko and W. B. Croft. Relevance based language models. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*,

pages 120–127. ACM Press, 2001.

[12] Movable Type Blacklist Filter, with content filtering, URL: http://www.jayallen.org/projects/mt-blacklist/.

[13] Joint statement from Yahoo, Google, and others regarding the "nofollow" tag, URLs: http://www.google.com/googleblog/2005/01/preventing-comment-spam.html, http://www.ysearchblog.com/archives/000069.html.

[14] No nofollow: fight spam, not blogs, URL: http://www.nonofollow.net.

[15] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 275–281. ACM Press, 1998.

[16] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz. A bayesian approach to filtering junk E-mail. In *Learning for Text Categorization: Papers from the 1998 Workshop*, Madison, Wisconsin, 1998. AAAI Technical Report WS-98-05.

[17] L. von Ahn, M. Blum, and J. Langford. Telling humans and computers apart automatically. *Commun. ACM*, 47(2):56–60, 2004.