Using Language Models for Spam Detection in Social Bookmarking

Toine Bogers and Antal van den Bosch

ILK, Tilburg University
P.O. Box 90153, 5000 LE
Tilburg, The Netherlands
{A.M.Bogers,Antal.vdnBosch}@uvt.nl

Abstract. This paper describes our approach to the spam detection task of the 2008 ECML/PKDD Discovery Challenge. Our approach focuses on the use of language models and is based on the intuitive notion that similar users and posts tend to use the same language. We compare using language models at two different levels of granularity: at the level of individual posts, and at an aggregated level for each user separately. To detect spam users in the system, we let the users and posts that are most similar to incoming users and their posts determine the spam status of those new users. We first rank all users in the system by KL-divergence of the language models of their posts—separately and combined into user profiles—and the language model of the new post or user. We then look at the spam labels assigned to the most similar users in the system to predict a spam label for the new user. We evaluate on a snapshot of the social bookmarking system BibSonomy made available for the Discovery Challenge. Our approach achieved an AUC score of 0.9784 on an internal validation set and an AUC score of 0.9364 on the official test set of the Discovery Challenge.

Key words: Social bookmarking, language modeling, spam detection, BibSonomy

1 Introduction

A prominent feature of the Web 2.0 paradigm is a shift in information access from local and solitary, to global and collaborative. Instead of storing, managing, and accessing personal information on only one specific computer or browser, personal information management and access has been moving more and more to the Web. Social bookmarking websites are clear cases in point: instead of keeping a local copy of pointers to favorite URLs, users can instead store and access their bookmarks online through a Web interface. The underlying application then makes all stored information sharable among users, allowing for improved searching and generating recommendations between users with similar interests.

Any system that relies on such user-generated content, however, is vulnerable to spam in one form or another. Indeed, many other electronic systems that allow users to store, share, and find online resources have also come under attack from spamming attempts in recent years. Search engines, for instance, suffer increasingly

from so-called *spamdexing* attempts with content especially created to trick search engines into giving certain pages a higher ranking for than they deserve [1]. Spam comments are also becoming an increasingly bigger problem for websites that allow users to react to content, like blogs and video and photo sharing websites [2].

Social websites and social bookmarking services have been becoming an increasingly popular part of the Web, but their focus on user-generated content also makes them vulnerable to spam, threatening their openness, interactivity, and usefulness [3]. Motivation for spamming can range from advertising and self-promotion to disruption and disparagement of competitors. Spamming is economically viable because the barrier for entry into the abused systems is generally low and because it requires virtually no operating costs beyond the management of the automatic spamming software. In addition, it is often difficult to hold spammers accountable for their behavior.

Spam for social bookmarking is a growing problem and this has been acknowledged by making it one of the tasks of the 2008 ECML/PKDD Discovery Challenge, along with tag recommendation. In this paper, we focus on the spam detection task alone. Our approach to spam detection is based on the intuitive notion that spam users will use different language than 'legitimate' users when posting resources to a social bookmarking system. We detect new spam users in the system by first ranking all the old users in the system by the KL-divergence of the language models of their posts—separately and combined into user profiles—and the language model of the new user or post. We then look at the spam labels assigned to the most similar users in the system to predict a spam label for the new user.

The paper is structured as follows. We start off by reviewing the related work in the next section, followed by a description of the task and the data set, our preprocessing steps, and our evaluation setup in Section 3. In Section 4, we discuss our approach to the spam detection task. Our results are presented in Section 5. We discuss our findings and conclude in Section 6.

2 Related Work

Spam issues in social bookmarking services have received relatively little attention so far. Heymann et al. (2007) examined the relationship between spam and social bookmarking in detail and classified the anti-spam strategies commonly in practice into three different categories: *prevention*, *detection*, and *demotion* [3]. *Prevention-based* approaches are aimed at making it difficult to contribute spam content to the social bookmarking system by restricting certain types of access through the interface (such as CAPTCHAs) or through usage limits (such as post or tagging quota).

Spam detection methods try to identify likely spam either manually or automatically, and then act upon this identification by either deleting the spam content or visibly marking it as such for the user [3]. To our knowledge, the only published effort of automatic spam detection for social bookmarking comes from Krause et al. (2008) who investigated the usefulness of different machine learning algorithms and features to automatically identify spam [4]. They tested their algorithms on a

data dump of the BibSonomy system. This data set was not the same as the one used in the 2008 Discovery Challenge and contained many features not available for the Discovery Challenge task.

Demotion-based strategies, finally, focus on reducing the prominence of content likely to be spam. Rank-based methods, for instance, try to produce orderings of the system's content that are both more accurate and more resistant to spam [3]. A demotion-based strategy for combating spam is described by Heymann et al. (2007) and described in more detail in Koutrika et al. (2007). They constructed a simplified model of tagging behavior in a social bookmarking system and compared different ranking methods for tag-based browsing. They investigated the influence of various factors on these rankings, such as the proportion and behavior of spam users and tagging quota [5], and found that ranking methods that take user similarity into account are more resistant to manipulation.

If we cast our nets a bit wider than just social bookmarking, we can find more anti-spam approaches in related fields, such as blogs. Mishne et al. (2005) were among the first to address the problem of spam comments in blogs and used language model disagreement between the blog post itself, the comments, and any pages linked to from the comments to identify possible spam comments [2]. In 2006, the TREC Blog Track also paid attention the problem of blog spam [6].

Finally, the data set for the 2008 Discovery Challenge is based on the BibSonomy social bookmarking service and, in addition to spam detection and tag recommendation, more research has been done using this system. See [4] for a short overview of the related work.

3 Methodology

3.1 Task description

We include a brief description of the spam detection task and the data in this section to allow this paper to be self-contained. The goal of the spam detection task of the Discovery Challenge was to automatically detect spam users in the provided snapshot of BibSonomy. The goal was to learn a model that can predict whether a user is a spammer or not. An added requirement was that the model should make good predictions for initial posts made by new users, in order to detect spammers as early as possible. This decision to identify spam at the user level—instead of at the post level—means that all of a user's posts are automatically labelled as spam. This decision was justified earlier[1] in Krause et al. (2008) by the observation that users with malicious intent often attempt to hide their motivations with non-spam posts [4]. In addition, Krause et al. also cite workload reduction as a reason for the decision to classify at the user level.

[1] Krause et al. are also the organizers of the 2008 Discovery Challenge, hence the same justification applies. Unfortunately, it is not clear if the results reported in their 2008 paper were achieved on the same data set as the one made available for the Discovery Challenge.

3.2 Data

For the spam detection task a snapshot was made available of the BibSonomy system as a MySQL dump, which consisted of all resources posted to BibSonomy between its inception and March 31, 2008. Two types of resources are present in the data set: bookmarks and BibTeX records. The training data set contained flags that identify users as spammers or non-spammers. The Discovery Challenge organizers were able to collect data of more than 2,400 active users and more than 29,000 spammers by manually labeling users. These labels were included in the data set for training and tuning parameters. Table 1 shows some simple statistics of the data set and illustrates the skewness present in the data set, both in terms of spammers and 'legitimate' users, and looking at the strong preference for bookmarks among spammers and BibTeX among legitimate users.

Table 1. Statistics of the BibSonomy data set.

	count
resources	14,074,956
bookmark, spam	13,257,519
bookmark, clean	596,073
BibTeX, spam	1,240
BibTeX, clean	220,124
users	31,715
spam	29,248
clean	2,467
average posts/user	59.8
spam	55.6
clean	108.9
tags	424,963
spam	69,902
clean	379,888
average tags/post	7.5
spam	8.2
clean	3.0

As mentioned before, two types of resources can be posted: bookmarks and BibTeX records, the latter with a magnitude more metadata available. In our approach we decided to treat BibTeX records and bookmarks the same and thus use the same format to represent them both. We represented all resource metadata in an TREC-style SGML format using 4 fields: `<TITLE>`, `<DESCRIPTION>`, `<TAGS>`, and `<URL>`. For the bookmarks, the title information was taken from the `book_description` field, whereas the `title` field was used for the BibTeX records. The `<DESCRIPTION>` field was filled with the `book_extended` field for bookmarks, whereas the following fields were used for the BibTeX records: `journal`, `booktitle`, `howPublished`, `publisher`, `organization`, `description`, `annote`, `author`, `editor`, `bibtexAbstract`, `address`,

school, series, and institution. For both resource types all tags were added to the <TAGS> field. The URLs extracted from the book_url and url fields were pre-processed before they were used: punctuation was replaced by whitespace and common prefixes and suffixes like www, http://, and .com were removed. Figure 1 shows an example of an instance of our XML representation.

```
<DOC>
  <DOCNO> 694792 </DOCNO>
  <TITLE>
    When Can We Call a System Self-Organizing
  </TITLE>
  <DESCRIPTION>
    ECAL Carlos Gershenson and Francis Heylighen
  </DESCRIPTION>
  <TAGS>
    search agents ir todo
  </TAGS>
  <URL>
    springerlink metapress openurl asp genre article issn 0302 9743
    volume 2801 spage 606
  </URL>
</DOC>
```

Fig. 1. An example of one of the posts (#694792) in our SGML representation.

Other than the data present in the provided data set, we did not use any other, external information, such as, for instance, the PageRank of the bookmarked Web page.

3.3 Evaluation

To evaluate our different approaches and optimized parameters, we divided the data set up into a training set of 80% of the users and a validation set of the remaining 20%. We evaluated our approaches on this validation set using the standard measures of AUC (area under the ROC curve) and F-score, the harmonic mean of precision and recall, with β set to 1. We optimized k using AUC rather than F-score, as AUC is less sensitive to class skew than F-score [7], and the data is rather skewed with 12 spam users for every clean user. For the final predictions on the official test set we used all of the original data as training material.

4 Spam Detection

4.1 Language Models for Spam Detection

Our approach to spam detection is based on the intuitive notion that spam users will use different language than legitimate users when posting resources to a social bookmarking system. By comparing the language models of posts made by spammers and posts made by legitimate users, we can use the divergence between the

models as a measure of (dis)similarity. After we have identified the k most similar posts or users using language modeling, we classify new users as spam users or genuine users by scoring these new users by how many spam posts and how many clean posts were found to be similar to it.

Language models [8] are a class of stochastic n-gram models, generally used to measure a degree of surprise in encountering a certain new span of text, given a training set of text. The core of most language models is a simple n-gram word prediction kernel that, based on a context of two or three previous words, generates a probability distribution of the next words to come. Strong agreement between the expected probabilities and actually occurring words (expressed in perplexity scores or divergence metrics) can be taken as indications that the new text comes from the same source as the original training text. Language models are an essential component in speech recognition [9] and statistical machine translation [10], and are also an important model in information retrieval [11]. In the latter context, separate language models are built for each document, and finding related documents to queries is transformed into ranking documents by the likelihood, estimated through their language model, that each of them generated the query.

In generating document language models, there is a range of options on the granularity level of what span of text to consider a document. At the most detailed level, we can construct a language model for each individual post, match these to the incoming posts, and use the known spam status of the best-matching posts already in the system to generate a prediction for the incoming posts or users. We can also take a higher-level perspective and collate all of a user's posts together to form large documents that could be considered 'user profiles', and generate language models of these individual user profiles. Incoming posts or users can then be matched against the language models of spammers and clean users to classify them as being more similar to one or the other category. Figure 2 shows how these two levels of language models relate to one another.

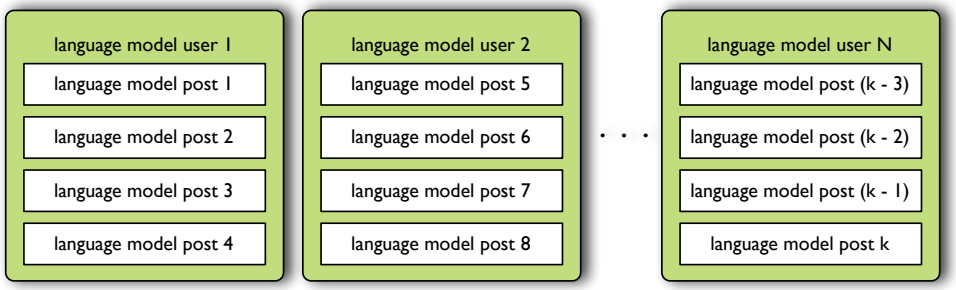

Fig. 2. Two types of language models: the models of the individual posts and the models of the user profiles.

A third option—at an even higher level of granularity—would be to only consider two language models: one of all spam posts and one of all clean posts. However, we believe this to be too coarse-grained for accurate prediction, so we did not pursue this further. Another extension to our approach would have been to use language models for the target Web pages or documents such as proposed by [2]. However, it is far from trivial to obtain the full text of all the source documents linked to by the BibTeX posts. Crawling all the target Web pages of the 2.2 million bookmark posts is impractical as well. Furthermore, we suspect that incorporating language models from all externally linked Web pages and documents would slow down a real-time spam filtering system to an undesirable degree.

We used the Kullback-Leiber divergence metric to measure the similarity between the language models. The KL-divergence measures the difference between two probability distributions Θ_1, Θ_2 is

$$KL(\Theta_1||\Theta_2) = \sum_w p(w|\Theta_1) \log \frac{p(w|\Theta_1)}{p(w|\Theta_2)} \quad (1)$$

where $p(w|\Theta_1)$ is the probability of observing the word w according to the model Θ_1 [2, 8].

The Indri toolkit[2] implements different retrieval methods based on language modeling. We used this toolkit to perform our experiments and construct and compare the language models of the posts and user profiles. The language models we used are maximum likelihood estimates of the unigram occurrence probabilities. We used Jelinek-Mercer smoothing to smooth our language models, which interpolates the language model of a post or user profile with the language model of background corpus, which in our case is the training collection of posts or user profiles. We chose Jelinek-Mercer smoothing because it has been shown to work better for verbose queries than other smoothing methods such as Dirichlet smoothing [12].

We performed two different sets of experiments. First, we compared the language models of the user profiles in our validation set with the language models of the profiles in our training set. For each test user profile we obtained a ranked list of best-matching training users. In addition, we did the same at the post level by comparing the test post language models with the language models of the training posts. Here, ranked lists of best-matching posts were obtained for each test post. These similarity rankings were normalized, and used as input for the spam classification step described in the next subsection.

For both the user and the post level we used all of the available fields—title, description, tags, and tokenized URL—to generate the language models of the posts and user profiles in our training collection. For the new posts and user profiles in our validation and test sets, however, we experimented with selecting only single fields to see what contribution each field could make to the spam detection process. This means that we have four extra sets of representations of the incoming validation and test documents, each with information from only one of the fields, bringing our total of representations for each of the two levels to five.

[2] Available at http://www.lemurproject.org

4.2 Spam Classification

After we calculated the language models for all posts and user profiles, we obtained the normalized ranking of all training documents, relative to each test post or user profile. For each of the best-matching training documents, we used the manually assigned spam labels of 0 or 1 to generate a single spam score for the new user. The simplest method of calculating such a score would be to output the spam label of the top-matching document. A more elegant option would be to take the most common spam label among the top k hits. However, we settled on calculating a weighted average of the similarity scores multiplied by the spam labels, as preliminary experiments showed this to outperform the other options. In the rare case that no matching documents could be retrieved, we resorted to assigning a default label of no spam (0) for these 0.7% of test users, as in our training set 84.2% of these unmatched users were not spammers. For post-level classification, this meant we obtained these weighted average spam scores on a per-incoming-post basis. To arrive at user-level spam scores, we then matched each incoming post to a user and calculate the average per-post score for each user.

One question remains: how many of the top matching results should be used to predict the spam score? In this, our approach is similar to a k-nearest neighbor classifier, where the number of best-matching neighbors k determines the prediction quality. Using too many neighbors might smooth the pool from which to draw the predictions too much in the direction of the majority class, while not considering enough neighbors might result in basing too many decisions on accidental similarities. We optimized the optimal value for k for all of the variants separately on the AUC scores.

5 Results

Table 2 lists the results of our different spam detection approaches. At the user level, the validation set representation where we only used the tags to construct our language models surprisingly outperformed all other approaches and representations, including the one with metadata from all fields. It achieved an AUC score of 0.9784 and the second highest F-score of all user-level representations at 0.9767. Our submission to the Discovery Challenge task was therefore made using this approach. The second best approach compared the language models of user profiles that used all metadata fields and achieved an 0.9688 AUC score. Overall, using the user-level language models outperformed the post-level language models.

Figures 3 and 4 shows the ROC curves for the 10 different combinations of fields and matching level. One surprising difference between the post-level and the user-level experiments is that at the user level the representation with only the tags works best, while it performs worst at the post level. Another interesting difference between post- and user-level experiments is the difference in the optimal number of nearest neighbors k. Matching users appears to require a considerably greater number of neighbors than arriving at a spam classification using only individual posts' language models.

Table 2. Spam detection results of the two approaches on the validation set. The optimal neighborhood sizes k were optimized on AUC scores. Best scores for each metric and level are printed in bold.

Level	Fields	Precision	Recall	F-score	AUC	k
user-level	all fields	**0.9659**	**0.9986**	**0.9820**	0.9688	180
	title	0.9534	0.9909	0.9718	0.9308	140
	description	0.9543	0.9976	0.9755	0.9228	95
	tags	0.9580	0.9961	0.9767	**0.9784**	195
	url	0.9502	0.9311	0.9406	0.8478	450
post-level	all fields	0.9735	**0.9950**	**0.9842**	**0.9571**	50
	title	0.9664	0.9815	0.9739	0.9149	50
	description	0.9707	0.9416	0.9559	0.8874	75
	tags	0.9804	0.7448	0.8465	0.7700	10
	url	**0.9773**	0.8940	0.9338	0.8730	15

Our submitted run on the test set provided by the Discovery Challenge used only the tags of each user to compare the language models with k set to 195. Classifying the incoming users as spammers or clean users achieved an AUC score of 0.9364. Precision was 0.9846, recall 0.9748, and the F-score 0.9797.

6 Discussion & Conclusions

In this paper we presented our language modeling approach to the spam detection task of the 2008 Discovery Challenge. We start by using language models to identify the best-matching posts or user profiles for incoming users and posts. We then look at the spam status of those best-matching neighbors and use them to guide our spam classification. Our results indicate that our language modeling approach to spam detection in social bookmarking systems shows promising results. This confirms the findings of [2], who applied a similar two-stage process using language modeling to detecting blog spam, albeit on a much smaller scale.

We experimented with matching language models at two different levels of granularity and found that, in general, matching at the user-level gave the best results. This was to be expected as the spam labels for the users in the data set were judged and assigned at the user-level. This means that the misleading, 'genuine' posts of spam users were automatically flagged as spam, thereby introducing more noise for the post-level matching than for the user-level matching.

The best performance at the user level was achieved by matching user-level language models using only the tags of the incoming users' posts. This is in line with the findings of [4], where the features related to the usage and content of tags were also found to be among the most important. Interestingly enough, matching posts only the incoming posts' tags resulted in the worst performance of all post-level runs. We can think of two likely explanations for this. The first is that the post-level approach is more likely to suffer from incoming posts without any assigned

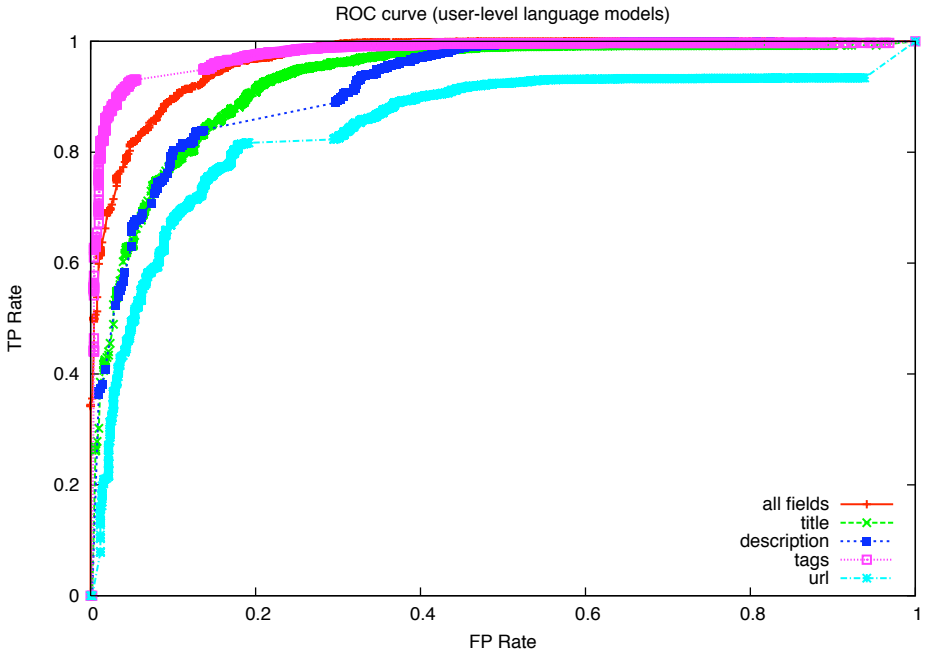

Fig. 3. ROC curve at the user level

tags than the user-level approach is. Although 99.95% of all posts in the data set have valid tags[3], this also means that it is possible for incoming posts to have no tags. Without any tags as metadata, our approach cannot find any matching posts in the system. At the user level, this is even less likely to happen: only 0.009% of all users never assign any tags. However, this might still be a valid reason to use all metadata fields for the user-level approach: with all available metadata we can increase coverage, because empty posts are not allowed by any social bookmarking system.

The second reason illustrates a possible limitation of our approach at the same time: spammers will change their behavior over time and might have done so in the time period the test set originates from. By generating metadata with a similar language model to the clean posts in the system, spammers could make it more complicated for our approach to distinguish between themselves and genuine users. However, this also makes it more difficult for the spammers themselves: it is very hard for a spammer to post resources to a social bookmarking system that will be both similar to existing posts and to the language of the spam entry. In addition, such behavior could easily be countered by extending our method to include the language models of the target resources or by including other features such as the PageRank of bookmarked pages. Extending our approach in such a way is

[3] Valid meaning with a tag other than `system:unfiled`.

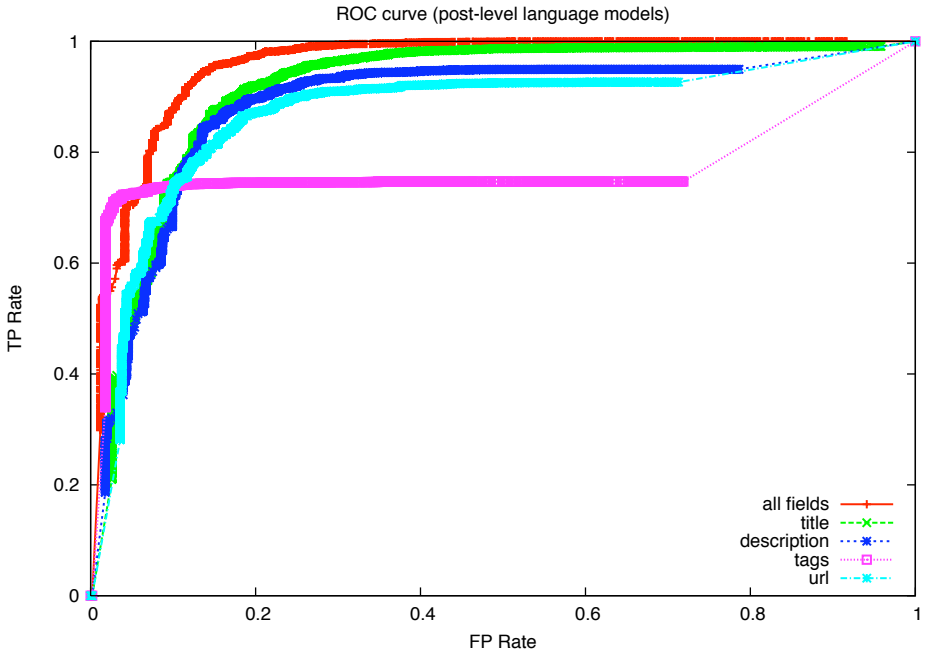

Fig. 4. ROC curve at the post level

one of the possible avenues for future work. Another would be to also restrict the language models of the training set to only certain fields and seeing how this influences performance. Finally, we would also like to test our approach on another social bookmarking system to see how our algorithms carry over to other systems.

One particular advantage of our approach is that it could be implemented with limited effort on top of an existing social bookmarking search engine. After any standard retrieval runs, the top k matching results can then be used to generate the spam classification, only requiring a lookup of predetermined spam labels.

As a final note we wish to briefly describe some of our experiences with applying language models to the other Discovery Challenge task of tag recommendation. By using a similar two-stage approach of first identifying similar posts and then aggregating the most popular tags associated with those best-matching posts, we were only able to achieve a maximum F-score of around 0.10. This clearly illustrates that an approach of finding the system-wide best matching posts for tag recommendation is not a good approach. We believe the reason for this to be that tagging, unlike spamming, is a much more personal activity: the tags another person assigned to the same resource need not necessarily be the tags a new user would apply. For spam detection our method only needs to assign one out of two possible labels to a new user, instead of picking 10 correct tags from a set of hundreds of thousands of possible tags for a new post. We therefore believe our language modeling approach to be better suited to the spam detection than to the tag recommendation.

Acknowledgments

The work described in this paper was funded by SenterNovem / the Dutch Ministry of Economics Affairs as part of the IOP-MMI À Propos project, and by the Netherlands Organization for Scientific Research as part of the NWO Vernieuwingsimpuls program.

Bibliography

[1] Gyöngyi, Z., Garcia-Molina, H.: Web Spam Taxonomy. In: AIRWeb '05: Proceedings of the 1st International Workshop on Adversarial Information Retrieval on the Web, Chiba, Japan (May 2005) 39–47

[2] Mishne, G., Carmel, D., Lempel, R.: Blocking Blog Spam with Language Model Disagreement. In: AIRWeb '05: Proceedings of the 1st International Workshop on Adversarial Information Retrieval on the Web, New York, NY, USA, ACM (2005) 1–6

[3] Heymann, P., Koutrika, G., Garcia-Molina, H.: Fighting Spam on Social Web Sites: A Survey of Approaches and Future Challenges. IEEE Internet Computing **11**(6) (2007) 36–45

[4] Krause, B., Hotho, A., Stumme, G.: The Anti-Social Tagger - Detecting Spam in Social Bookmarking Systems. In: AIRWeb '08: Proceedings of the 4th International Workshop on Adversarial Information Retrieval on the Web. (2008)

[5] Koutrika, G., Effendi, F.A., Gyöngyi, Z., Heymann, P., Garcia-Molina, H.: Combating Spam in Tagging Systems. In: AIRWeb '07: Proceedings of the 3rd International Workshop on Adversarial Information Retrieval on the Web, New York, NY, USA, ACM (2007) 57–64

[6] Ounis, I., de Rijke, M., McDonald, C., Mishne, G., Soboroff, I.: Overview of the TREC 2006 Blog Track. In: TREC 2006 Working Notes. (2006)

[7] Fawcett, T.: ROC Graphs: Notes and Practical Considerations for Researchers. Machine Learning **31** (2004)

[8] Manning, C.D., Schütze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge, MA (1999)

[9] Jelinek, F.: Self-organized Language Modeling for Speech Recognition. Readings in Speech Recognition (1990) 450–506

[10] Brown, P.F., Cocke, J., Della Pietra, S.A., Della Pietra, V.J., Jelinek, F., Lafferty, J., Mercer, R.L., Roossin, P.S.: A Statistical Approach to Machine Translation. Computational Linguistics **16**(2) (1990) 79–85

[11] Ponte, J.M., Croft, W.B.: 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, New York, NY, ACM Press (1998) 275–281

[12] Zhai, C., Lafferty, J.: A Study of Smoothing Methods for Language Models Applied to Information Retrieval. ACM Transactions on Information Systems **22**(2) (2004) 179–214