AI Report

Anirudh Ravi
IMT2013005

May 14, 2017

1 Abstract

Social networking sites offer users the option to submit user spam reports for a given message, indicating this message is inappropriate. In the paper[1] a framework that uses these user spam reports for spam detection is presented. The framework is based on the HITS web link analysis framework and is instantiated in three models. The models subsequently introduce propagation between messages reported by the same user, messages authored by the same user, and messages with similar content. Models were tested on data from a popular social network and were compared with one another, based on message content and raw report counts. Each of the models can also be extended to remove "untrustworthy" reporters by coming up with a threshold on the reporter scores. These extended models are found to be performing better than the existing models.

2 Introduction

Within social networks, spam can be found in publicly visible pages (groups, celebrity profiles, etc.), on profile pages, and in private inboxes. Most networking sites allow their users to issue a report on all of these levels when they feel a message is spam or otherwise inappropriate (e.g., violent or sexist language). These user spam reports are the main ingredient of this paper. The paper[1] explores the usefulness of user spam reports in classifying spam in social networks. More precisely, the paper[1] presents a framework that indicates the likelihood of a message being spam, based on user spam reports.
The framework tries to improve over the simple counting method by taking into account not only the number of spam reports, but also the spam reporters and the authors of the messages. The core assumption that underlies our framework is that spam messages are more likely to be spam when they have been reported as spam by several "trustworthy" users. We define a trustworthy user as one who issued spam reports for a large number of messages that are actually spam messages. An obvious way of formalizing this assumption is to use (a variation of) the HITS link analysis algorithm, in which hubs and authorities are used to propagate scores. The initial model is further refined by propagating spam

scores not only via spam reporters, but also via messages with similar content and messages written by the same author.

3 Spam Detection Framework

The spam detection framework introduced in the paper[1] is based on HITS and uses the links between messages and other objects to propagate spam scores. Three instantiations of our spam detection framework are introduced, each of which builds on the previous instance. The models assign spam scores to each message in the dataset, allowing us to rank messages.

3.1 Reporter Model

In the Reporter Model we interpret the problem space as a bipartite graph. The graph has two node types: (i) reporters and (ii) messages. Directed edges go from reporters to messages, indicating that the reporter issued a user spam report regarding this particular message.

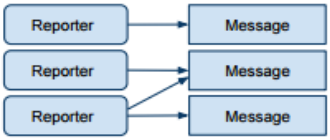


Fig. 1: Example graph of the Reporter Model.

The spam score S(m) for a given message node m in the graph is the sum of all hubs (i.e., reporters) that are connected to it:

$$S(m) = \sum_{r \in R_m} H(r),$$

where R_m represents the set of all reporter nodes that are connected to message node m. The report score H of a node r is the sum over all authorities (messages) connected to this node:

$$H(r) = \sum_{m \in M_r} S(m),$$

where M_r represents all messages that are connected to reporter node r. An intuitive interpretation for this model is that the hub score H(r) of a reporter r represents her trustworthiness. The authority (spam) score can thus be seen as a weighted version of a spam score based on raw report counts.

The final hub and authority scores are found using an iterative procedure. First, the reporter scores are initialized uniformly. The new message scores are calculated using the above spam score calculation equation and normalized. Based on the new message scores, we invoke reporter score calculation equation to calculate new reporter scores. These steps are repeated until convergence occurs, which is defined as the total change compared the previous iteration being constant.

3.2 Author-Reporter Model

The graph of the Reporter Model is extended by introducing a third node type: the author of a message. Each message has exactly one author, which is encoded using a directed edge from the author to the message. The reporter nodes and author nodes do not overlap, so even if the same user is the author of one message and reporter of another messages, it is still modeled using two different reporter and author nodes.

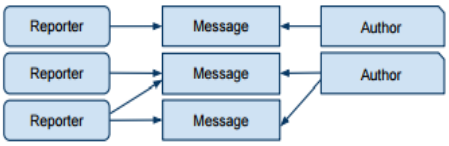


Fig. 2: Graph of the Author-Reporter Model.

Adding authors to the model leads to a change in the previous spam score equation, which now includes an author score: in which a_m denotes the author

$$S(m) = A(a_m) + \sum_{r \in R_m} H(r),$$

of message m. The author score A is similar to the reporter score H, except that we now sum over all messages written by author a (i.e., M_a).

$$A(a) = \sum_{m \in M_a} S(m),$$

in which M_a indicates all messages authored by author a. The intuition behind this score is that an author, who posted lots of spam messages, is more likely to post another spam message than a user who posted no spam messages at all.

3.3 Similarity-Author-Reporter Model

The final model also includes links between messages, that is, links between messages with similar content. We opt to model content similarity using cosine similarity and we include edges between a message and the n most similar messages, given that the similarity score is higher than zero.

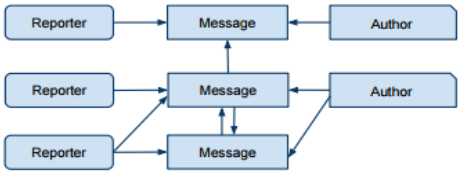


Fig. 3: Graph of the Similarity-Author-Reporter Model.

In this case, we calculate spam scores using

$$S(m) = (1-\gamma)[A(a_m) + \sum_{r \in R_m} H(r)] + \gamma \sum_{n \in I_m} S'(n),$$

where $S^{'}$ denotes the spam score from the previous iteration, I_m is the set of messages similar to message m, and γ is a free parameter.The intuition behind adding similar messages is that messages with similar content should also have a similar spam score, and therefore, spam scores should be propagated between those messages.

4 Dataset

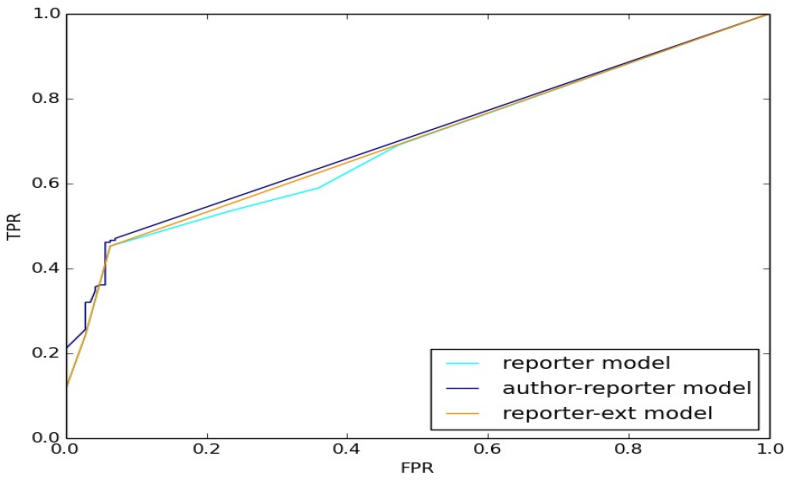
Our dataset consists of messages, spam reports, and users from the largest Dutch social networking site, Hyves. Each spam report concerns one single message on the profile page of a public figure or on a publicly accessible group page. The dataset consists of 28,998 spam reports, collected during the period from January 2010 to January 2011. The spam reports cover 13,188 unique messages and are generated by 9,491 unique reporters/users.
We find that the dataset is quite sparse. For the messages with at least one report, we find that by far most messages have just one user spam report (11,993 messages). About 750 messages have two user spam reports, 180 have three reports, 90 messages have four reports, etc. Only 38 messages in our dataset have 10 or more user spam reports. In a real-world scenario, the data is likely to be denser, which could result in an increase in performance.
The annotators—professional moderators working at Hyves—annotated 1,195 messages in total. Of these, 698 messages are marked as spam and 497 are marked as not spam.

5 Approach

Models are compared on their ability to rank messages by spam score, that is, push spam messages to the top of a spam ranking, and the messages that are not spam to the bottom. For presenting the results we use receiver operating characteristic (ROC) curves and the area under the curve (AUC). To plot the ROC curve for a model we have to find the TPR(true positive rate) and FPR(false positive rate) values.
I started off by implementing two of the three models, Reporter and Author-Reporter Model in python. The entire dataset is divided into 2 sets: training and testing. Applying the model on the dataset will result in two lists: sorted spam scores of all the messages and sorted reporter scores of all the reporters. First, a model is applied to the training dataset which gives us two lists: sorted spam scores associated to messages and sorted reporter scores associated to reporters. Similarly, the model is also applied to the testing dataset which also gives two lists.
We define a variable/threshold on the spam scores of the training set to say that the messages that have spam scores above threshold are spam otherwise not spam. After doing so, we iterate through the spam scores list of the testing set and based on that particular message's spam score we predict the message's class using the spam scores list of the training set(which is already divided into spam and ham classes based on the threshold). Since each message in the testing set already has an annotation associated to it(either spam or ham) four scenarios rise up:- we predict the message to be spam and the message is spam, we predict the message to be ham and the message is ham, we predict the message to be spam and the message is ham and the final one we predict the message to be ham and the message is spam. We then calculate the FPR and TPR values. The ROC curve is plotted by varying the thresold values on the spam scores list of the training set and find out the corresponding FPR and TPR values.
I extended the existing models by putting a threshold on the reporter scores of the reporters. Since reporter score is a measure of trustworthiness of a reporter then by putting a threshold on the reporter score we can remove all the untrustworthy reporters from the system. After removing the untrustworthy reporters we run the model again and it is found to perform slightly better than the existing model.

6 Results

AUC for unsupervised reporter model: 0.689980063027
AUC for unsupervised author-reporter model: 0.706138658435
AUC for unsupervised collusion-reporter model: 0.697247411409(threshold:-0.000000000005)



7 Conclusion

Currently, in the extended models the threshold value is found through trail and error process. In the future, I plan to work on finding a way to find the threshold value in a more procedural sense.

8 References

[1] Maarten Bosma, Edgar Meij, and Wouter Weerkamp. A Framework for Unsupervised Spam Detection in Social Networking Sites. In *Proceeding ECIR'12 Proceedings of the 34th European conference on Advances in Information Retrieval*, 2012.