Automatic Moderation of Online Discussion Sites

Abstract

Online discussion sites (ODS) are plagued with various types of unwanted content such as spam, obscene and malicious contents. Prevention and detection-based techniques have been proposed to filter inappropriate content (IC) out from ODS. But, while prevention techniques have been widely adopted, detection of IC remains mostly a manual task. Existing detection techniques, which are divided into rule-based and statistical techniques, suffer from various limitations. Rule-based techniques usually consist of manually crafted rules or blacklists of keywords. Both are time-consuming to create and tend to generate too many false-positives and false-negatives. Statistical techniques typically use corpora of labeled examples to train a classifier to tell "good" and "bad" messages apart. Although statistical techniques are generally more robust than rule-based techniques, they are difficult to deploy because of the prohibitive cost of manually labeling examples.

In this paper, we describe a novel classification technique to train a classifier from a partially labeled corpus and use it to moderate IC in ODS. Partially labeled corpora are much easier to produce than completely labeled corpora, as they are only made up with unlabeled examples and examples labeled with a single class (e.g. "bad"). We implemented and tested this technique on a corpus of messages posted on a stock message board and compared it with two baseline techniques. Results show that our method outperforms the two baselines and that it can be used to significantly reduce the number of messages that need to be reviewed by human moderators.

1 Introduction

Like many other social media sites, online discussion sites (ODS) are plagued with various types of unwanted content such as spam, or obscene and malicious messages. ODS encompasses a variety of systems that enable users to engage in conversations online or offline, publicly or privately. Examples of ODS are forums, commenting systems, review sites, chatrooms, etc. Characterized by a low editorial control, and offering amazing opportunities to engage with people and disseminate information quickly, ODS make an attractive target for spammers and other fraudsters. In addition, as the role of social media in the digital marketing mix of organizations continues to grow, they have also become more exposed to the publication of inappropriate content such as defamation, information leakage or stock ramping. According to the security firm Websense, spam and malicious comments alone account for 95% of all user-generated comments to blogs[1]. This paper addresses the problem of moderating ODS and it presents a novel technique for automatically identifying contributions not complying with a site's terms of use.

Prevention and detection-based techniques have been proposed to filter out inappropriate content (IC) from ODS. But, while prevention techniques, such as CAPTCHAs[2] or account fees have been widely adopted, detection of IC remains mostly a manual task. Indeed, existing detection techniques, which are divided into rule-based and statistical techniques, suffer

[1]http://investor.websense.com/releasedetail.cfm?ReleaseID=409218

[2]A CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) is a type of challenge-response test used in computing to ensure that the response is not generated by a computer.

from various limitations. Rule-based techniques, such as whitelists of senders or blacklists of keywords, usually consist of manually crafted rules [16]. For instance, Kontostathis et al. recently developed a rule-based system to detect cyberpredators in Internet Relay Chat [16]. One of the main problems with these techniques is that building and maintaining rules is generally time-consuming and laborious. Moreover, they tend to generate too many false-positives and false-negatives. In other words, they can easily consider a message containing objectionable keywords in an appropriate context as inappropriate or, conversely, they can consider a message containing malicious content expressed with acceptable terms as appropriate. Statistical techniques are usually based on machine-learning. They use corpora of labeled examples to train a classifier to separate "good" from "bad" messages [25, 34]. For instance, Pendar compared SVM with kNN classifiers to separate predator communication from victim communication based on the content of the messages [25]. Statistical techniques, like rule-based techniques, are often similar to spam identification techniques. But, although statistical techniques have been broadly adopted for spam filtering, they are seldom used to moderate ODS because of the lack of labeled examples and of the prohibitive cost of labeling a large-enough number of examples.

In this paper, we present a novel classification-based technique for moderating ODS that can train a classifier from a partially labeled corpus. Typically, a partially labeled data set is made up with a certain number of examples labeled as "bad" and a large amount of unlabeled data potentially containing both "good" and "bad" examples. Therefore, they are faster and cheaper to generate than completely labeled corpora required by other classification-based techniques. The technique has been implemented and compared with two baseline techniques in an evaluation on a large corpus of messages posted in a stock message board. Results show that our method outperforms the two baselines and that it can be used to drastically reduce the number of messages that need to be reviewed by human moderators. At this stage, the method can effectively be used in a semi-automated system for moderating IC but not in a completely automated system as the proportion of false-positive is still too significant. In the last section, we discuss some of the main challenges that need to be addressed to build a completely automated moderation system.

The remainder of the paper is organized as follows. Section 2 provides background information about inappropriate content in ODS and discusses the limitations of existing human-based approaches to moderation. Section 3 reviews the literature on automatic filtering of inappropriate content in ODS and overviews related work on classification from partially-labeled data. In Section 4, we present our approach and the implementation of our filtering technique. Section 5 describes the dataset used for our experiments. Experimental results are presented in Section 6, and finally, in Section 7, we give some concluding remarks and future work.

2 Background

Inappropriate content (IC) pervades online discussion sites such as forums, newsgroups and chatrooms. There are basically three main types of content that can be particularly damaging for a site's reputation and trust. The first is filthy content that includes material that is coarse, profane, obscene, pornographic, vulgar or blasphemous. The second type is spam. It comprises unsolicited messages sent in bulk, messages sent without permission, intended to promote/advertise products, and web spam. Web spam, or spam-indexing, is one of the most common type of spam in social media. It is meant to mislead search engines into ranking some pages higher than they deserve [10]. Last, malicious content includes material that is intended to hurt or deceive the website, its users or individuals/organizations. For instance, malicious content targeting users may be defamatory or misleading such as messages intended to spread false rumors or viruses. Malicious content distributed via social networking sites tends to be particularly effective in terms of successful infection. According to Internet security vendor Kaspersky Lab, infection rate of malicious code distributed via social networking sites is ten times greater than infection rate via e-mail[3]. To deal with the threat represented by IC, various tools and techniques have been proposed.

Heymann et al. [13] distinguish three main types of techniques to deal with IC in social media: prevention, demotion and detection. Prevention techniques attempt to make contribution of spam more difficult by changing interfaces or limiting user actions for example through CAPTCHAs or account fees. Demotion techniques attempt to make ranking algorithms resistant to inappropriate content, or, in other words, they try to lower the ranking of inappropriate content (IC) for example by taking into account user ratings or author's popularity. Last, detection techniques attempt to identify IC, and remove it or reduce its prominence. Although prevention techniques can help reduce IC, they cannot eliminate it.

[3]http://www.kaspersky.com/news?id=207575818

In addition, as demotion techniques are mostly intended to ranking systems, detection techniques are the main approaches to fight IC in online discussion sites.

Most existing websites rely on manual moderation to detect IC. In the classic approach, user contributions are moderated by a group of special users called *moderators*. These users have moderation privileges that allow them to accept/reject and delete messages from other users. Moderators intervene either before or after publication of messages. In the case of *pre-moderation*, all submissions remain pending until they are approved by the moderators. The advantage of this approach is that all published material has been moderated; the main drawback is that the publication delay can be long due to moderation delays (MD), thus refraining users from using the service. Therefore, most systems implement a *post-moderation* strategy, where submissions are immediately published but can be verified and removed any time after publication. Post-moderation enables the removal of publication delays, but it has no impact on moderation delays. Moreover, the proportion of published contributions that have been verified, or moderation coverage (MC), is often small in case of post-moderation.

Alternative strategies to centralized moderation have been designed to reduce MD and increase MC using community feedback. For example, in a distributed moderation system, all users are given moderation privileges to enable the community to self-moderate. Reactive moderation system is a hybrid approach between centralized and distributed moderation where moderation is performed by moderators, but the user community has the ability to alert them when they find IC. Although reactive and distributed moderation enable the verification of more messages and more quickly, in a system with a large number of contributions, long MD and small MC are unavoidable. In the next section, we overview approaches for automatically moderating message platforms the advantages of which are to moderate all messages and drastically reduce MD.

Detection techniques for ODS, whether they are rule-based or statistical can use four different types of features to identify IC. In this section, we start by presenting how these features are used by filtering techniques. We then discuss the limitations of filtering techniques based on classification, and we conclude by an overview of existing classification techniques based on partially-labeled data.

2.1 Automatic filtering of IC in ODS

Filtering techniques can be distinguished into four basic types: origin-base, link-based, collaborative-based and content-based techniques.

Origin-based filters determine the legitimacy of a message with respect to the credibility of its source [9, 33]. For instance, observing that many Twitter[4] accounts of spammers frequently use a combination of letter and figures, Yardi et al. [33] have built a spam detection system that reacts to that type of pattern. Commercial systems such as Defensio[5] and Akismet[6] which provide filtering services accessible through APIs, rely on various author information, such as author's URL, email and IP. To determine the spaminess of a message, this information is typically crossed with information about sender and content reputation

[4]http://www.twitter.com

[5]http://defensio.com

[6]http://akismet.com

provided by projects such as McAfee's TrustedSource[7] or the Spamhaus Project[8]. The main problem with this approach is that a trusted source may occasionally submit inappropriate content. Further, these filters may be deceived by malicious users who start by posting appropriate messages before publishing malicious content.

Link-based techniques use hyperlinks to identify inappropriate content. For instance, TrustRank uses the link structure of the web to address the web spam problem [11]. The algorithm starts from a list of seed pages manually evaluated by an expert as spam (bad) and ham (good). Then, an algorithm identifies other pages that are likely to be ham based on their connectivity with the good seed pages. Link-based techniques are mostly useful for spam detection, and they cannot be used to detect IC in messages that do not contain hyperlinks.

Collaborative-based approaches rely on users' feedback to separate inappropriate from legitimate content. The main limitation of these approaches is the sparsity problem, i.e., if a message has not been previously reviewed, it is not possible to determine whether the message is appropriate or not. In social websites, users often have the possibility to rate others' contributions. Several works have proposed techniques to automatically predict the quality of a contribution based on their content and the ratings they have received. Collaborative approaches are current with reputation systems where objects (e.g.. news or products) are ranked based on reviews given by user community. For instance, Hsu et al. [14] propose a machine learning-based approach for ranking comments based on the community's preferences. The goal of this approach is to promote high-praised comments and filter out

[7]http://www.trustedsource.org

[8]http://www.spamhaus.org

low-praised comments. One of the main drawbacks of this approach is that it is vulnerable to fake reviews intended to push up (ballot stuffing) or down (bad mouthing) the ranking of an object [2, 3, 30].

Content-based filters resort to natural language processing techniques to identify IC. Blacklists of keywords are the most common rule-based techniques. Recently, Kontostathis et al. developed a rule-based system to detect cyberpredators in Internet Relay Chat [16]. They developed a dictionary and a codebook to recognize various constructs characteristic of luring communication. The luring terms, words, icons, phrases, and netspeak contained in the dictionary are used by explicit rules and categorization instructions defined in the coding manual. One of the main problems with these techniques is that building and maintaining rules is generally time-consuming and laborious. Moreover, they tend to generate too many false-positives and false-negatives. Statistical techniques bear strong resemblance with statistical anti-spam filters which have been shown to perform effectively for that task [8, 24, 28]. However, the literature of statistical techniques for moderating IC in ODS is scarce. Recently, Pendar compared SVM with kNN classifiers to separate predator communication from victim communication based on the content of the messages [25]. Yin et al. addressed the problem of detecting harassment in online communities using text classification [34]. They propose a supervised learning approach employing local features, sentiment features, and contextual features of user contributions. Sentiment features are helpful to estimate a level of animosity in a message, while contextual features are helpful to estimate the topic deviation of a message with respect to previous messages in the thread.

Although statistical techniques have been broadly adopted for spam filtering, they are seldom used to moderate ODS because of the lack of labeled examples and of the prohibitive cost of labeling a large-enough number of examples. It is often infeasible to obtain moderation labels for IC in ODS. Indeed, in practice, moderators only label inappropriate messages that they discover. Unlabeled messages may be either appropriate content or inappropriate content that has not yet been verified by moderators. Therefore, real-world moderation corpora, when available, usually only contain partially labeled examples. Techniques have been proposed to train a classifier from partially labeled data sets that consist in datasets having some examples labeled in one-class (e.g., "inappropriate") and a large amount of unlabeled examples. In the next subsection, we review some of these techniques.

2.2 Learning from partially-labeled data

The objective of a classification technique is to classify a given instance into one of the pre-defined classes with minimum error. In the moderation corpora, pre-defined classes are "inappropriate content" and "appropriate content". Typically, a classification technique first builds a model based on a set of *instances* (or *examples*) labeled with the relevant classes and subsequently uses this model to classify new instances.

Most of the existing classification techniques require a large number of labeled examples for each of the pre-defined classes in order to achieve high prediction accuracy. However, in real world applications it is often very expensive and infeasible to obtain such large amount of fully labeled data. In recent years, several techniques have been proposed to deal with this problem of lack of availability of fully labeled data. One approach that addresses this problem is called *semi-supervised learning.* Semi-supervised learning is useful in situations where a limited amount of labeled examples exists for every class along with large amount of unlabeled examples. It has been shown that under certain conditions, using unlabeled data in conjunction with the labeled data can increase the accuracy of the classifier [**?**, 23].

As they require at least a limited amount of labeled data for each of the pre-defined classes, semi-supervised classifiers are not applicable to data such as the moderation corpora which contains labeled instances only for one class, namely the *"inappropriate content"*. In applications where we focus only on one class, the class of interest is usually referred to as the *positive* class and everything else is taken as the *negative* class. For example, in a spam filtering corpus, positive instances would be instances labeled as spam. Datasets that consists of labeled instances only for the positive class along with a large amount of unlabeled instances are known as *partially labeled* data. In recent years, researchers have investigated approaches to deal with such partially labeled data which do not have any labeled instances for the negative class. The problem of *learning from partially labeled data* is also known as *learning from positive and unlabeled data* [21], or *single-class learning* [32]. There are two major approaches.

The first approach is a two-step strategy such as Roc-SVM [21] and PEBL [35]. In the first step, reliable negative instances are identified from the unlabeled data using some heuristics. Roc-SVM uses Rocchio, a well-known query-reformulation technique [27], while PEBL uses a technique called 1-DNF which is based on disjunctive normal forms. Then, in the second step, a classifier is built using the given positive instances and negative instances identified in the first step, and improved by means of a bootstrapping technique. Both Roc-SVM and PEBL use Support Vector Machines (SVM) in this step [31]. Finally, one model is selected as the final learning model. Roc-SVM uses a selection process to select the best classifier as the final model, while PEBL chooses the last one. The advantage of the two-step approach is that, at the end of the first step, there are labeled instances for both positive and negative classes, and hence any of the existing classification techniques can be applied in the second step. However, the accuracy of the final classifier largely depends on the quality of the negative instances extracted in the first step.

The second approach tries to build a model based on estimates of the proportion of positive instances and the effect of positive and negative instances in the unlabeled data. Existing models usually consist in Naive Bayes classifiers using custom estimates [7, 12, 36]. Given a set of labeled instances for every pre-defined class in the data, a Naive Bayes classifier builds the learning model based on the probability estimates for each of the pre-defined classes [19]. Hence, it is infeasible to build a learning model using a Naive Bayes classifier in the absence of labeled instances for the negative class. PNB [7] overcomes this problem by using an estimate of the positive class probability, expected to be provided by the user. Using this estimate of the positive class probability p and taking the negative class probability as 1-p, gives the prior distribution of the model. The major disadvantage of this method is that it expects the user to provide the positive class probability. This is hard in practice in real world applications. To overcome this limitation, several PNB-based techniques have been proposed that automatically estimate the positive class probabibility [12, 36]. For example, [36] propose a method called prTFIDF that is largely based on PNB. However, in their approach, the probability is estimated by maximizing a performance function proposed in [17] specifically designed to evaluate the performance of PPU classifiers, which only depends on the accurate prediction of positive examples (unlike other conventional measures such as, for example, the F-score).

In this paper, we present an approach which does not require any negative class estimates. This overcomes the problem of identifying reliable negative instances as in the first approach. It also eliminates the need for the positive class probability estimate as in the second approach. We discuss our methodology in detail in the following section.

3 Our approach - Positive and Unlabeled data Learning based on Lift (PULL)

3.1 Overview

Our approach is based on a measure of correlation, known as the *Interest factor* [29] or *Lift* [1]. Given a vocabulary of words V, a set of positive documents p, and a set of unlabeled documents u, the *Lift* L measures the probability of a subset of words $v \in V$ occurring in p against the probability of occurring in the entire sample.

$$L = \frac{P(v|p)}{P(v)} \tag{1}$$

If L is greater than 1, the probability of v occurring in the positive class is higher than that of the entire sample. Hence, the occurrence of v is considered to be positively correlated to the positive class. On the other hand, if L is less than 1, it suggests that the occurrence of v is negatively correlated to the positive class. If L is equal to 1 then the occurrence of v is independent of the class of the document. This can be summarized as follows:

$$\text{L} \begin{cases} > 1, & \text{if occurrence of } v \text{ is positively correlated to the positive class} \\ = 1, & \text{if occurrence of } v \text{ is independent of the class of the document} \\ < 1, & \text{if occurrence of } v \text{ is negatively correlated to the positive class} \end{cases} \tag{2}$$

In other words, if v is positively correlated to p

$$P(v|p) > P(v) \tag{3}$$

Let $v = \{v_1, v_2, ...v_k\}$. As in Naive Bayes classifier [20], assuming that the occurrence of a word v_i is statistically independent of the occurrence of any other word in v, we can re-write condition 3 as:

$$\prod_{i=1}^{k} P(v_i|p) > \prod_{i=1}^{k} P(v_i) \tag{4}$$

Condition 4 forms the basis of our approach. Given a document in the form of a word vector, we calculate the probability of the occurrence of the given word vector in the positive class (left-hand side of condition 4), and the probability of occurrence in the entire data (right-hand side of condition 4). If the LHS is greater than the RHS, the document is predicted to be positive. Otherwise, it is predicted to be negative.

As discussed in the previous section, the major advantage of our approach is that it does not need any estimate of the negative class distribution for the class prediction.

3.2 Implementation

The key step in implementing our approach is to calculate the word probabilities for the positive class $P(v_i|p)$, and for the entire sample $P(v_i)$. As in a Naive Bayes classifier, we estimate the word probabilities by counting the word occurrences in the respective class as follows:

$$P(v_i|p) = \frac{N(v_i, p)}{N(p)} \tag{5}$$

$$P(v_i) = \frac{N(v_i, n)}{N(n)} \tag{6}$$

where n is the set of all instances, p is the set of labeled positive instances, $N(v_i, x)$ is the number times the word v_i occurs in the set x, and $N(x)$ is the total number of word occurrences in set x. Assuming that the unlabeled data is generated according to the underlying generative model, and the word distribution in the unlabeled positive documents is the same as the labeled positive documents, Equation 6 can be re-written as:

$$P(v_i) = \frac{N(v_i, u)}{N(u)} \tag{7}$$

where u is the set of unlabeled instances.

Re-writing the prediction condition of our approach given by 4, using Equations 5 and 7, we obtain:

$$\prod_{i=1}^{k} \frac{N(v_i, p)}{N(p)} > \prod_{i=1}^{k} \frac{N(v_i, u)}{N(u)} \tag{8}$$

As in a Naive Bayes classifier, we need to apply Laplace correction in order to avoid condition 8 resulting in zero, when a word frequency is zero in given set. Hence, in our implementation we use the following formula:

$$\prod_{i=1}^{k} \frac{N(v_i, p) + e}{N(p) + e * |v|} > \prod_{i=1}^{k} \frac{N(v_i, u) + e}{N(u) + e * |v|} \tag{9}$$

where $|v|$ is the number of words from the vocabulary that occur in the given instance, and e is a constant. e is usually set to 1, however, for highly imbalanced classes this value tends to biaised towards the minority class. Hence, in our experiments, we set e to 0.5.

3.3 Discussion

Similarly to PNB [7] and PrTFIDF [36], PULL is based on the Naive Bayes classification model. However, it differs from these approaches for two main reasons. First, PULL does not need an estimate of the proportion of positive instances in the labeled data (p). In [7], this probability is supposed to be given while in [36], the probability is estimated by maximizing a performance function. Using a different performance measure would give a different estimate. Second, PULL uses a different technique to estimate the probability of a feature in the negative class $P(v|N)$. In PULL, $P(v|N)$ is estimated by $P(v)$ while PNB and PrTFIDF use $1-p$ to estimate it. In PNB and PrTFIDF, the estimate of $P(v|N)$ may have negative values because of the way it is calculated. The greater the estimate p is, the greater the probability of $P(w|N)$ to be negative is. To overcome this problem, the estimate of $P(v|N)$ needs to be smoothen which may significantly alter the natural distribution of $P(v|N)$, consequently affecting the classifier accuracy.

4 Data and Preprocessing

4.1 Data

The data for this study comes from HotCopper, Australia's largest internet stock message board. The time period for our study runs from January 2008 to December 2008 inclusive. Messages were downloaded from the HotCopper website using software written by the authors, and we restricted our collection of data to only include messages that have a ticker symbol field representing an ASX listed company published in the "ASX By Stock" forum. Each message contains the following fields: date, time, author, ticker symbol, content. In total, the data set contains $1,146,223$ messages about $1,825$ firms listed on the Australian

Securities Exchange (ASX).

< Table 1 comes here >

In accordance with Australian law regarding all information available in the securities market, HotCopper users are expected to comply with a set of strict usage guidelines[9]. Messages published on the forum are post-moderated by a team of moderation volunteers and forum administrators[10]. In addition, forum users are able to alert moderators if they find inappropriate content (as in reactive moderation).

An interesting feature of moderated posts is that they are labeled with the main reason of their moderation. Moderated posts are not removed from the forum, but their content is replaced by a message which contains the following information: moderation time, moderation type, and a comment specifying the reasons for moderation. The full set of moderation types with their respective percentage composition is given in Table 1; each type is a word or a phrase which describes the primary reason for moderation. Some examples of textual comments are provided below:

- All you seem to be doing is plaguing threads with your downramping on non held stocks. Find something else to do with your time instead of wasting others. Snide remarks on stocks are not considered helpful or useful posts. (moderation type is Ramping)
- Flaming? Profanity? Defamatory? Baiting? Gosh, I am spoilt for moderating choice!

[9]User Posting Guidelines, http://www.HotCopper.com.au/postingguidelines/

[10]List of moderators, http://www.HotCopper.com.au/forum_modList.asp

How about you have a good think about how you would like to post in future ... Thanks ... PD (moderation type is Other)

We obtained the original content of moderated posts, which is not publicly available on the website, directly from HotCopper.

4.2 Preprocessing

Web content in general and Web 2.0 content in particular tend to be noisy textual data. Forum posts for instance contain texts written in a SMS-style or abbreviations or misspelled terms or phonetic expressions. Therefore, pre-processing techniques are generally required to prepare the text before its features can be used by a classifier. Our preprocessing approach aims at cleaning the data and abstracting some features in order to simplify the data. Data abstraction targets links and numerical values. Links are very common in spam messages. In order to take into account the impact of a link in a message on its probability to be inappropriate, we replace all string patterns representative of links by a token "LINK". When possible, numerical values are also replaced by a token indicating the type of data. A common problem when trying to detect profanity messages is voluntary misspelling of profanity words in order to deceive blacklist-based filters. A common technique is to replace one or two characters, for instance: "sh*t", "sh!t". A possible solution to detect this feature is to run a spelling correction system. Here, we replace non alphabetical characters contained in words by "_". Eventually, too frequent words are discarded using a stoplist[11]. The following

[11]We use NLTK english stoplist: http://www.nltk.org

actions are performed on all messages in this order:

1. text is lowercased,
2. HTML tags are deleted,
3. monetary values are replaced by “MONETARY”,
4. percentage values are replaced by the token “PERCENT”,
5. other numbers are replaced by the token “NUMBER”,
6. urls are replaced by the token “LINK”,
7. non alphabetical characters contained in words are replaced by “_”,
8. common words listed in a stoplist are discarded, and
9. only words of sizes comprised between 3 and 20 are kept for further processing, and other tokens are discarded.

As previously discussed, HotCopper posts may be moderated due to several reasons, but each moderated post can only be labeled with one of the moderation types given in Table 1. This restriction and the close similarity between some of the moderation types may result in ambiguities that will affect the classifier performances. For example, a post which could be labeled as both ‘Unlicensed Advice’ and ‘Ramping’ may only be labeled as ‘Unlicensed Advice’. Posts that are moderated due to multiple reasons are often labeled as ‘Other’ (as shown in the previous section). The ‘Other’ class also contains posts submitted on the wrong forum, posts submitted by multiple account users (multinic), etc. As this class has no characteristic features, it is not possible for a classifier to learn to identify posts that should be moderated as Other. Therefore, we discard posts in this class when training and testing the classifiers. Due to the close similarity between 'Other' and 'Unknown Reason', we also discard posts labeled with 'Unknown Reason'. Finally, we also discard 'Duplicate' and 'Copyright', two other classes that cannot be predicted solely from the content. Duplicate corresponds to identical posts that have been posted by the same author, while 'Copyright' includes posts that contains material which infringes copyright. These types of posts could be more easily identified with techniques for detecting plagiarism and document overlap [22, 15].

5 Experiment

5.1 Performance Measures

< Table 2 comes here >

Traditionally, the performance of a classification system is evaluated based on its overall accuracy which is the percentage of instances that are correctly classified by the system. However, in the case of data such as moderation copora where one class (moderated posts) is extremely small compared to the other class (unlabeled posts), the overall accuracy becomes inadequate. For example, consider a dataset with 99% instances representing the majority class and only 1% instances representing the minority class. In this case, even if all the instances are classified as majority class, the overall accuracy will be very high (99%). However the minority class is usually the class of interest and, in this case, none of these instances are correctly classified. Hence, when using such corpora with imbalanced classes, it is important to evaluate the predictive accuracy of the minority class. Typically in such corpora the minority class is considered to be the *positive* class, and the *True Positive Rate* or the *Recall* of the positive class is regarded as one of the important performance measures. Given a confusion matrix as in Table 2, the Positive Recall, which is the proportion of correctly classified positive instances, is given by:

$$Recall_p = \frac{TP}{TP + FN}$$

While it is important to have a high recall for the positive class, that alone is not sufficient to have a good classifier. For example if all the instances are classified as positive, the $Recall_p$ will be 100%. However, all the negatives (majority class instances) will be misclassified as positive. Therefore, in order to balance the predictive accuracy in both classes, a classifier should have high recall for both the positive and negative classes, where negative recall is given by:

$$Recall_n = \frac{TN}{TN + FP}$$

Robert et al. [26] use the geometric mean of $Recall_p$ and $Recall_n$ as performance measure.

$$g - mean = \sqrt{Recall_p \times Recall_n}$$

Another performance measure that is commonly used to evaluate classification task is the *F-Measure*. F-Measure is the harmonic mean of Precision and Recall which are defined by:

$$Precision_p = \frac{TP}{TP+FP} \quad Recall_p = \frac{TP}{TP+FN}$$

$$\text{F-Measure} = \frac{2 \times Precision_p \times Recall_p}{Recall_p + Precision_p}$$

It can be seen that F-measure balances the trade-off between Precision and Recall.

Usually, the reported F-measure is the average of the F-measures of the different classes. In the case of datasets with imbalanced classes, the average F-measure is biased either towards the majority or the minority class, depending on the averaging function. Therefore, in our experiments, we report the *g-mean* as the measure to compare of the performance of the classifiers.

5.2 Experimental Set-up

We conducted experiments to compare the performance of our classifier with two other classifiers, PNB [7] and Roc-SVM [21], which represent each of the two approaches for single-class learning described in Section 2.2. We implemented the PNB algorithm using Java and used the executable of Roc-SVM downloaded from the author's website[12].

In each experiment, we used 60% of the data for training the model and the remaining 40% for testing. We applied the preprocessing technique described in section 5.2 on the data

[12]http://www.cs.uic.edu/ liub/LPU/LPU-download.html

and represented posts as ‘bags-of-word’. Since the dataset contains extremely large number of instances (1,146,223 posts), the ‘bag of words’ approach resulted in millions of features. This high dimensional feature space is a major problem in text classification tasks. Most of the features in this high dimensional space are just noise and do not provide any useful information to the classification model. Hence, in our experiments we reduced the feature space by using only the words that occurred at least in one of the positive documents.

5.3 Step 1: Identifying moderated posts

< Table 3 comes here >

Our first experiment was to build a classifier to identify the instances that need to be moderated. The results of this experiment are given in Table 3. It can be seen that our classifier results in much higher geometric-mean than PNB and Roc-SVM. In particular, our positive recall is significantly higher than the other two classifiers. Positive recall gives the percentage of the positive instances identified by the classifier. Hence, a high value of positive recall indicates the ability of the classifier to identify the inappropriate posts.

The negative recall of our classifier is lower than that of PNB and Roc-SVM, Roc-SVM achieving the highest negative recall among the three. It is important to note that the negative recall should be interpreted in a different way when using partially labeled data. Generally, the negative recall represents the percentage of correctly classified negative instances. However, in partially labeled data, we do not have any instances labeled as negative. We only have instances labeled as positive and a large set of unlabeled data. The unlabeled data consists of positive as well as negative instances. Therefore, in partially labeled data, the negative recall represents the percentage of unlabeled data that are classified as negative. The best negative recall is the actual percentage of negative instances in the unlabeled data and cannot be 100% when there are positive instances among unlabeled data.

In this experiment, we built a single classifier to identify the posts moderated due to any of the several moderation causes given in Table 1. The major problem with this approach is that it is not possible to explain the cause of moderation for a post classified as 'moderated'. We overcome this problem in the next experiment where we build an individual classifier for each of these causes.

5.4 Step 2: Explaining moderated posts

HotCopper moderators are expected to explain why a post has to be moderated. A moderation explanation includes a moderation cause ('Ramping', 'Flaming', . . .) and a moderation comment. To assist moderators in that task, we compared two approaches for linking moderation predictions to moderation causes. These approaches are therefore run on posts that have been classified as positive during the first step. Both approaches rely on a set of specialized classifiers that can predict if a post belongs to a specific class.

< Table 4 comes here >

In the ‘Multiple classifiers’ approach, a classifier is trained to identify each moderation cause. Table 4 reports the results for this approach. The performances vary significantly according to the labels. The best predicted labels are ‘Flaming’, ‘Profanity’ and ‘Defamatory’. Generally, the larger the number of instances used for training, the better the classifier performs. Even though ‘Ramping’ has one of the largest number of instances, its classifier does not perform as well as other classes with a large number of instances. Ramping posts containing malicious content intended to manipulate market prices by influencing other forum users’ trading decisions. For instance, posts containing expressions such as “I heard from a credible source…” or “There is a speculation in the market…” have been labeled as ‘Ramping’ by HotCopper moderators. Successful attempts of market manipulation through ramping in stock message boards have been reported in the literature [18], and their impact on the market prices has recently been empirically demonstrated [6]. The difficulty of identifying posts labeled as ‘Ramping’ stems from the fact that it relies more on semantic and contextual features than the other classes.

< Table 5 comes here >

The goal of the ‘Multiple clustered classifiers’ approach is to increase the number of instances in each class, and to reduce the overlap between the classes, by using fewer labels. Instances labeled either as ‘Unlicensed Advice’, ‘Ramping’ or ‘Insider Trading’ are relabeled as ‘Cluster ramping’. Similarly, ‘Flaming’, ‘Profanity’, ‘Defamatory’, ‘Sexist’, ‘Racist’, and ‘Blasphemous’ are merged into ‘Cluster flaming’, and ‘Spamming’ and ‘Advertising’ are merged into ‘Cluster spamming’. Posts containing words in their comment indicating that they could belong to a different cluster are also assigned to that cluster. A list of keywords associated to each cluster is used to label posts from moderation comments. Table 5 reports the results for this approach. The geometric mean for the 'Cluster ramping' and 'Cluster flaming' classes is comparable with the geometric-mean for the 'Ramping', and 'Flaming' classes with the previous approach. Although this approach does not improve the classification for any particular label, except for spam, it produces a more coherent set of labels with a reduced overlap between the classes.

6 Discussion and Conclusion

In this paper, we have presented a novel classification technique to train a classifier from a partially labeled corpus and use it to moderate IC in ODS. We implemented and tested this technique on a corpus of messages posted on a stock message board and compared it with two baseline techniques. Results show that our method outperforms the baselines, and it can effectively be used to identify messages that are likely to contain inappropriate material. Thus, the method can be used in a semi-automated context to support the moderation task by reducing the number of messages that need to be reviewed by moderators. However, the number of misclassification errors remains too significant for the method to be used in a completely automated context.

The fact that a high number of acceptable messages are classified as inappropriate is not specifically due to the proposed classification technique. Most messages are misclassified because they contain features which are wrongly considered as representative of the positive class. Indeed, our current attribute selection technique consists in selecting words occurring in at least one of the instances in the positive class. This results in too many words being selected and words being improperly considered as significantly discriminant of the positive class. In addition, the method also has some cases of false-negative the number of which could be reduced with a better attribute selection technique. For example, some messages are acceptable on the surface (i.e. lexically) but their content may be harmful to forum users. A possible solution to address this problem is to rely on contextual features that can be extracted from the same thread as the message. The following examples show user reactions to messages that are likely to contain inappropriate content.

- *Drunk your an idiot....keep ramping and see what happens. (...)*
- *Puket So stop manipulating the SP from where you are (...)*
- *oman Your comments are the type I take offence at (...)*

Contextual features may be very useful to semantically characterize the content of a target message using lexical processing [5]. In future work, our main goal is to address these issues by coming up with a better attribute selection technique for partially labeled data.

References

[1] M. J. Berry and G. Linoff. *Data Mining Techniques For Marketing, Sales and Customer Support*. John Wiley and Sons, 1997.

[2] R. Bhattacharjee and A. Goel. Avoiding ballot stuffing in ebay-like reputation systems. In *P2PECON '05: Proceedings of the 2005 ACM SIGCOMM workshop on Economics of peer-to-peer systems*, pages 133–137, New York, NY, USA, 2005. ACM.

[3] R. Burke, B. Mobasher, R. Bhaumik, and C. Williams. Collaborative recommendation vulnerability to focused bias injection attacks. In *Proceedings of the Workshop on Privacy and Security Aspects of Data Mining, held at ICDM'05, Houston, Texas, USA*, 2005.

[4] O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.

[5] J.-Y. Delort. Automatically characterizing salience using readers' feedback. *Journal of Digital Information*, 10(1), 2009.

[6] J.-Y. Delort, B. Arunasalam, M. Milosavljevic, and H. Leung. The impact of manipulation in internet stock message boards. In submission, 2010.

[7] F. Denis, R. Gilleron, and M. Tommasi. Text classification from positive and unlabeled examples. In *Proceedings of the 9th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, IPMU*, 2002.

[8] H. Drucker, V. Vapnik, and D. Wu. Support vector machines for spam categorization. *IEEE Transactions on Neural Networks*, 10(5):1048–1054, 1999.

[9] H. Esquivel, A. Akella, and T. Mori. On the effectiveness of IP reputation for spam filtering. In *COMSNETS: Proceedings of the Second International Conference on COMmunication Systems and NETworkS*. IEEE, 2010.

[10] Z. Gyongyi and H. Garcia-Molina. Web spam taxonomy. Technical Report 2004-25, Stanford InfoLab, March 2004.

[11] Z. Gyöngyi, H. Garcia-Molina, and J. Pedersen. Combating web spam with trustrank. In *VLDB '04: Proceedings of the Thirtieth international conference on Very large data bases*, pages 576–587. VLDB Endowment, 2004.

[12] J. He, Y. Zhang, X. Li, and Y. Wang. Naive bayes classifier for positive unlabeled learning with uncertainty. In *SDM*, pages 361–372, 2010.

[13] P. Heymann, G. Koutrika, and H. Garcia-Molina. Fighting spam on social web sites: A survey of approaches and future challenges. *IEEE Internet Computing*, 11:36–45, 2007.

[14] C.-F. Hsu, E. Khabiri, and J. Caverlee. Ranking comments on the social web. In *CSE '09: Proceedings of the 2009 International Conference on Computational Science and Engineering*, pages 90–97, Washington, DC, USA, 2009. IEEE Computer Society.

[15] J. W. Kim, K. S. Candan, and J. Tatemura. Efficient overlap and content reuse detection in blogs and online news articles. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 81–90, New York, NY, USA, 2009. ACM.

[16] E. L. Kontostathis A and L. A. Chatcoder: Toward the tracking and categorization of internet predators. In *Proceedings of Text Mining Workshop 2009 held in conjunction with the Ninth SIAM International Conference on Data Mining (SDM 2009)*, 2009.

[17] W. S. Lee and B. Liu. Learning with positive and unlabeled examples using weighted logistic regression. In *ICML*, pages 448–455, 2003.

[18] D. J. Leinweber and A. N. Madhavan. Three hundred years of stock market manipulations. *The Journal of Investing*, 10(2):7–16, 2001.

[19] D. D. Lewis. Naive (bayes) at forty: The independence assumption in information retrieval. In *Proceedings of ECML-98, 10th European Conference on Machine Learning*, pages 4–15. Springer Verlag, Heidelberg, DE, 1998.

[20] D. D. Lewis. Naive (bayes) at forty: The independence assumption in information retrieval. In *ECML '98: Proceedings of the 10th European Conference on Machine Learning*, pages 4–15, London, UK, 1998. Springer-Verlag.

[21] X. Li and B. Liu. Learning to classify texts using positive and unlabeled data. In *IJCAI'03: Proceedings of the 18th international joint conference on Artificial intelligence*, pages 587–592, San Francisco, CA, USA, 2003. Morgan Kaufmann Publishers Inc.

[22] K. Monostori, A. Zaslavsky, and H. Schmidt. Document overlap detection system for distributed digital libraries. In *DL'00: Proceedings of the fifth ACM conference on Digital libraries*, pages 226–227, New York, NY, USA, 2000. ACM.

[23] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2-3):103–134, 2000.

[24] A. Ntoulas, M. Najork, M. Manasse, and D. Fetterly. Detecting spam web pages through content analysis. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 83–92, New York, NY, USA, 2006. ACM.

[25] N. Pendar. Toward spotting the pedophile telling victim from predator in text chats. In *ICSC '07: Proceedings of the International Conference on Semantic Computing*, pages 235–241, Washington, DC, USA, 2007. IEEE Computer Society.

[26] M. K. Robert, R. Holte, and S. Matwin. Learning when negative examples abound. In *Proceedings of ECML-97, Lecture Notes in Artificial Intelligence*, pages 146–153. Springer Verlag, 1997.

[27] J. J. Rocchio. Relevance feedback in information retrieval. In G. Salton, editor, *The SMART retrieval system: Experiments in automatic document processing*, pages 313–323. Englewood Cliffs, NJ: Prentice Hall, 1971.

[28] 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.

[29] P. Tan, V. Kumar, and J. Srivastava. Selecting the right interestingness measure for association patterns. In *Proceedings of the Eight ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2002.

[30] B. Van Roy and X. Yan. Manipulation-resistant collaborative filtering systems. In *RecSys '09: Proceedings of the third ACM conference on Recommender systems*, pages 165–172, New York, NY, USA, 2009. ACM.

[31] V. N. Vapnik. *The Nature of Statistical Learning Thoery.* Springer-Verlag NewYork Inc., 1995.

[32] C.-P. Wei, H.-C. Chen, and T.-H. Cheng. Effective spam filtering: A single-class learning and ensemble approach. *Decision Support Systems*, 45(3):491 – 503, 2008. Special Issue Clusters.

[33] S. Yardi, D. Romero, G. Schoenebeck, and d. boyd. Detecting spam in a twitter network. *First Monday [Online]*, 15(1), 12 2009.

[34] D. Yin, Z. Xue, L. Hong, B. D. Davison, A. Kontostathis, and L. Edwards. Detection of harassment on web 2.0. In *Proceedings of the 1st Content Analysis in Web 2.0 Workshop*, 2009.

[35] H. Yu, J. Han, and K. C. Chang. Pebl: positive example based learning for web page classification using svm. In *KDD '02: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 239–248. ACM Press, 2002.

[36] D. Zhang and W. S. Lee. A simple probabilistic approach to learning from positive and unlabeled examples. In *International Conference on Advanced Data Mining and Applications*, pages 118–129, 2005.

List of tables and figures

Moderation cause	Total	Total %
Other	6,379	45.1%
Flaming	3,583	25.3%
Ramping	1,519	10.7%
Profanity	1,207	8.5%
Spamming	389	2.8%
Defamatory	351	2.5%
Unknown Reason	159	1.1%
Advertising	152	1.1%
Unlicensed Advice	72	0.5%
Duplicate	73	0.5%
Insider Trading	73	0.5%
Copyright	63	0.4%
Sexist	43	0.3%
Racist	43	0.3%
Blasphemous	33	0.2%
Total	14,139	

Table 1: Moderation types found on HotCopper in 2008

	Actual Positive	Actual Negative
Predicted Positive	TP	FP
Predicted Negative	FN	TN

Table 2: Confusion Matrix

	Training	Test	PULL		PNP		Roc-SVM	
Pos class	(P/U)	(P/U)	(P/N)	GM	(P/N)	GM	(P/N)	GM
Moderated	(6372/708082)	(3641/472694)	(68/72)	70	(18/98)	42	(29/95)	52
Combined	(6372/708082)	(3641/472694)	(68/72)	70	(18/98)	42	(29/95)	52

This table reports the results of the experiment where the positive class is the set of ‘moderated’ posts and the negative class is the set of ‘appropriate’ posts. In the table:

(P/U) - Number of positive examples used / Number of unlabeled examples used

(P/N) - $Recall_p/Recall_n$

GM - $geometric\_mean = (Recall_p \times Recall_n)^{1/2}$

Table 3: Identification of moderated posts

	Training	Test	PULL		PNP		Roc-SVM	
Pos class	(P/U)	(P/U)	(P/N)	GM	(P/N)	GM	(P/N)	GM
Flaming	(2402/712052)	(1367/474968)	(76/75)	75	(20/99)	44	(26/97)	50
Ramping	(896/713558)	(646/646)	(53/71)	61	(0/100)	0.3	(5/99)	22
Profanity	(810/713644)	(491/475844)	(73/76)	74	(11/99)	33	(15/99)	39
Spamming	(302/714152)	(120/476215)	(9/96)	29	(3/100)	17	(0/99)	0.3
Defamatory	(256/714198)	(109/476226)	(63/90)	75	(10/99)	31	(0/99)	0.3
Advertising	(112/714342)	(58/476277)	(26/92)	49	(7/99)	26	(7/99)	26
Unlicensed Advice	(48/714406)	(26/476309)	(15/83)	32	(0/99)	0.3	(0/100)	0.3
Insider Trading	(41/714413)	(34/476301)	(32/84)	52	(3/99)	17	(0/100)	0.3
Sexist	(24/714430)	(22/476313)	(32/87)	53	(0/100)	0.3	(0/100)	0.3
Racist	(26/714428)	(21/476314)	(29/89)	51	(0/100)	0.3	(0/100)	0.3
Blasphemous	(26/714428)	(10/476325)	(50/85)	65	(0/100)	0.3	(0/100)	0.3

This table reports the results of a set of experiments performed with the positive class as the individual causes of moderation and the negative class is all the examples that do not belong to the positive class. In the table:

(P/U) - Number of positive examples used / Number of unlabeled examples used

(P/N) - $Recall_p/Recall_n$

GM - $geometric\_mean = (Recall_p \times Recall_n)^{1/2}$

Table 4: Multiple classifiers

	Training	Test	PULL		PNP		Roc-SVM	
Pos class	(P/U)	(P/U)	(P/N)	GM	(P/N)	GM	(P/N)	GM
Cluster ramping	(1246/713208)	(884/475451)	(58/67)	62	(0.3/99)	5	(7/99)	26
Cluster flaming	(3899/710555)	(2215/474120)	(76/73)	74	(24/98)	48	(32/96)	55
Cluster spamming	(433/714021)	(192/476143)	(15/94)	38	(4/100)	20	(3/99)	17

This table reports the results of a set of experiments performed by grouping similar causes of moderation into a single positive class and the negative class is all the examples that do not belong to the positive class. In the table:

(P/U) - Number of positive examples used / Number of unlabeled examples used

(P/N) - $Recall_p/Recall_n$

GM - $geometric\_mean = (Recall_p \times Recall_n)^{1/2}$

Table 5: Multiple clustered classifiers