On Free Speech and Civil Discourse: Filtering Abuse in Blog Comments

D. Sculley
Tufts University, Department of Computer Science
161 College Ave., Medford, MA 02155 USA
dsculley@cs.tufts.edu

Abstract

Internet blogs provide forums for discussions within virtual communities, allowing readers to post comments on what they read. However, such comments may contain *abuse*, such as personal attacks, offensive remarks about race or religion, or commercial spam, all of which reduce the value of community discussion. Ideally, filters would promote civil discourse by removing abusive comments while protecting free speech by not removing any comments unnecessarily. In this paper, we investigate the use of *user flags* to train filters for this task, with the goal of empowering each community to enforce its own standards. We find encouraging results on experiments using a large corpus of blog comment data with real users flags. We conclude by proposing several novel deployment schemes for filters in this setting.

1 Introduction

Internet *blogs* – online journals – have become important forums for fostering community discussion. In a typical blog, readers may write *blog comments* in response to a blog posting. While the majority of blog comments are not abusive, some comments do contain abuse. Unlike email spam, blog comment abuse is not primarily commercial in nature. More often, comment abuse contains personal attacks, obscenities, and even messages of hate based on race, religion, or nationality. Such comments mar the ability of blogs to foster constructive discussion in a virtual community.

1.1 User Flags and Community Standards

Typical blog services allow the community to protect itself from abusive comments through the process of user *flagging*, in which readers are given the option to flag abusive comments. Comments receiving a sufficient number of flags are removed from view. This flagging process enables a virtual community to enforce its own standards for civility in discourse. Unfortunately, this system, abusive comments must be seen and read to be flagged. Thus, an abusive comment may still negatively impact the community, and abusers may re-post comments after flagging.

Thus, we propose that automated filters, similar to email spam filters, be trained with user flag information. This allows filters to enforce standards of civil discourse set by the community – so long as free speech is protected by maintaining low false positive rates. However, flags are *noisy*. Users may flag inconsistently, inaccurately, or even maliciously. Thus, care is needed to construct and evaluate filters capable of learning from this data.

1.2 Contributions

Our investigation centers on a multi-lingual corpus comments with associated user flag information which is three orders of magnitude larger than the data set used in the largest prior study we are aware of. To our knowledge, this is the first reported use of user flags for training blog comment abuse filters. We show that, despite noise, user flag data can train filters that approach the performance of dedicated human annotators. Additionally, this paper gives analysis of blog comment abuse, compares several filtering methods, and offers suggestions for practical application.

2 Related Work

To our knowledge, relatively little work on blog comment abuse filtering appears in the literature, and none on using user flag data for training. However, there is significant prior work in email spam filtering and splog detection. We review work in these related fields here.

	cricket	getahead	money	movies	news	sports	total
unique blogs	416	188	1,380	627	1,748	405	4,764
unique userids	28,051	7,967	42,339	44,940	61,138	8,262	130,885
total comments	101,662	13,545	124,597	188,222	497,312	22,524	947,862
flag rate	0.12	0.07	0.11	0.22	0.23	0.19	0.20

Table 1: **Summary statistics for the `msgboard1` corpus of blog comments, broken out by topic.**

2.1 Email Spam Filtering

A wide variety of machine learning methods have been applied for online filtering of email spam, including variants of Naive Bayes [9, 15], perceptron algorithm variants [22], logistic regression [8], support vector machine (SVM) variants [21], compression-based methods [1], and ensemble methods [14] to name just a few. This approach has proven remarkably effective in laboratory evaluations such as TREC [3], with best methods routinely performing above the 0.999 level for the Area under the ROC curve (ROCA) measure.

There are important differences between filtering email spam and blog comment abuse. In the TREC-style tests for email spam filtering, it is assumed that all of the training and data has gold-standard quality labels [5] that are equivalent to a consistent, accurate human judgment for each new message. In contrast, the training labels for blog comment abuse are given by user flags from thousands of users, who may be inconsistent or even malicious in their flagging.

Another difference is that blog comments are read by a *community* of readers, rather than an individual email recipient, and abuse is defined by the standards of the community. While email spam is motivated primarily by commercial intent [6], the majority of this blog comment abuse appears to be socially motivated. Thus, abusive blog comments are most often unique. With the exception of relatively infrequent commercial blog comment abuse, campaigns of abusive comments similar to high-volume email spam campaigns are rare.

2.2 Splog Detection

A related task involving filtering and blogs is *splog* detection. A splog (or, *spam blog*) is a fake blog intended to fool search engines into assigning undue importance to an associated website [11]. This is done by inserting links from the splog to the website, increasing the site's PageRank (an importance measure based on link structure). The *splog detection* problem is important to search engines, and has been approached content analysis [11] and temporal and link analysis [13].

This task is essentially distinct from the task of filtering blog comment abuse. In the abuse filtering task we are concerned with removing obscene, offensive, or commercial comments from real blogs rather than detecting fake blogs. These tasks do overlap, however, in relatively rare cases when abusers insert links within comments to deceive search engines.

2.3 Blog Comment Abuse Filtering

To our knowledge, the only prior work on blog comment abuse filtering centered on a small, hand labeled data set of roughly one thousand examples [16]. Mishne *et al.* proposed filtering blog comments by measuring the disagreement between language models for comments and associated blog entries.

In this paper, we do not consider language-model disagreement methods for two reasons. First, it has been shown that SVM variants exceed the performance of the language-model disagreement method on the same data set, using information only from the comment [21]. Second, although our multi-lingual corpus contains comments primarily in English, many comments are either partly or completely written in other languages. This renders language-model disagreement methods problematic.

3 The `msgboard1` Data Set

This paper centers on the `msgboard1` corpus, a data set of nearly one million blog comments with user flag information (see Table 1). The corpus of blog comments was provided by Rediff.com, a leading blog hosting service catering to the needs of India's large expatriot community.[1]

The corpus consists of blog comments gathered from January through August of 2007. The corpus contains comments from blogs on six self-identified topics, listed in Table 1. There are comments from a total of 4,764 unique blogs, of which 2,901 contribute at least ten comments to the corpus. Although the primary language of the comments is English, there are also comments written either partly or entirely in Hindi, Tamil and many other languages of India, all represented with the standard ASCII character set. Misspellings and character substitutions are common.

[1]This corpus may be available on a per-case basis for research purposes. Contact pranshus@rediff.co.in.

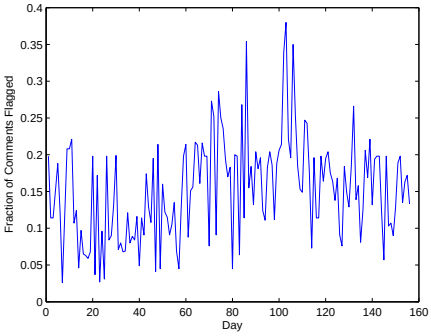

Figure 1: **Flag rates over time for the most popular blog.** The spikes indicate periods of high amounts of flagging, often caused by abusive *flame wars* among users. Graphs for other blogs show similar patterns.

Each comment is annotated with a `userid` identifying the author of the comment, a `blog` title showing the blog in which it was posted, and a `flag` variable showing whether or not the comment had been flagged by users. Over half of the 130,855 unique `userid`'s contributed only a single comment, rendering user history insufficient for reliable filtering.

3.1 Patterns of Abuse

Are comments flagged at a steady rate, or are there flagging peaks and lows? To answer this, we plotted flag rate (fraction of comments getting flagged) per day for the most popular blogs over the span of data collection, smoothing the rates to remove the effect of low-traffic days. Figure 1 shows results for the most popular blog (with over 40,000 comments); results for other high-volume blogs were similar. Interestingly, the graph shows several aperiodic spikes, denoting dates in which a large percentage of comments were flagged.[2] We speculate that these spikes are caused primarily by *flame wars*, often seen in blog comments, in which users direct abuse at each other in heated conflict.

3.2 Understanding User Flags

What causes a comment to get flagged? To explore this question, we computed information gain [23] values for words in comments that have been flagged, and for non-flagged comments. (A selection of these are shown in Table 2.) Brief examination of each case showed that obscenities and commercial terms ranked highly for flagged comments, which agrees with the intuitive notion of comment abuse. Religious terms also ranked highly for flagged comments, highlighting the large amount of abuse found in the corpus that contains hateful messages directed against members

[2]It is possible that these temporal patterns may be useful in abuse filtering, an area for future work.

FLAGGED		NON-FLAGGED	
`com`	wife	india	right
prophet	dog	people	please
contest	lovers	good	best
hmm	launched	think	same
annihilate	sexual	country	agree
ipods	lord	time	money
women	alternative	mr	life
causing	bangladesh	team	president
defending	hassle	world	work

Table 2: **Selected words with high information gain, for flagged and non-flagged comments.** Obscenities and references to specific religious figures have been removed from the flagged list for display, and stop words have been removed from the non-flagged list.

of several different religions. For non-flagged terms, it was surprising to note that common stop words (such as pronouns and articles) scored highly – perhaps because comments written with proper grammatical usage tended not to be abusive. Other words that stand out are terms associated with civil discourse.

To further understand the breakdown of abuse, we examined a random uniform sample of 100 flagged comments. In this sample, 39 were found to be contain obscenities or personal attacks, 39 were found to contain racial, national, or religiously motivated comments, 9 were found to be of commercial intent, and the remaining 12 were flagged for other reasons. We were surprised at both the high amount of socially motivated abuse, together totaling almost 80% of the flags, and also at the relatively low amount of commercial abuse.

4 Online Filtering Methods

For our experiments, we treat the blog comment abuse filtering task as an *online filtering* task, to model the effect of filtering a stream of incoming comments over time. This is the same scenario as been widely applied in email spam filtering evaluations [6]. In this scenario, the filter is shown one comment a time, in a time-ordered sequence where i is the current time step. Each comment is represented by a feature vector $\mathbf{x}_i \in \mathbb{R}^n$, with an associated label $y_i \in \{-1, +1\}$ for unflagged and flagged comments, respectively. For each example (comment), the filter is asked to predict *abuse* or *non-abuse* using a function $f(\mathbf{x}_i)$, with the user-flag information hidden. Once the prediction is made, the user-flag information y_i is revealed to the filter, which may then update its prediction model $f(\cdot)$ as needed using $(\mathbf{x}_i, y_i)$.

In this paper, we evaluate the performance of several online machine learning methods for abuse fil-

tering, each of which has been successfully applied in email spam filtering. Two of these are variants of the Naive Bayes classifier, which represents the *generative* methodology. In this approach, the filter models the joint probability $p(\mathbf{x}, y)$ to help compute the posterior probability $p(y|\mathbf{x})$ [18]. The remaining classifiers follow the *discriminative* approach of learning a mapping from $\mathbf{x}$ to y directly [18].

4.1 Naive Bayes

Variants of Naive Bayesian classifiers have been widely applied in email spam filtering [9, 15]. Metsis *et al.* recently compared several Naive Bayes variants, and found the multinomial Naive Bayes variant using Boolean features to be most effective for email spam [15]. We use this form of Naive Bayes in our experiments, testing two different priors.

For notation, let each feature in $\mathbb{R}^n$ be referred to with a unique identifier t_j where $1 \leq j \leq n$. Let the elements of example $\mathbf{x}_i \in \mathbb{R}^n$ be referred to as $\{\mathbf{x}_{i1}, ..., \mathbf{x}_{in}\}$, containing values in $\{0, 1\}$, and let t_{jy} refer to the number of times feature t_j occurs in examples with label y: that is, $\sum_{k=1}^{i} \mathbf{x}_{kj}(y_k = y)$.

Following Metsis [15], the classification function for multinomial Naive Bayes is:

$$f(\mathbf{x}_i) = \frac{p(y=+1) \cdot \Pi_{j=1}^{i} p(t_j|y=+1)^{\mathbf{x}_{ij}}}{\sum_{y \in \{-1,+1\}} p(y) \cdot \Pi_{j=1}^{i} p(t_j|y)^{\mathbf{x}_{ij}}}$$

and we predict *abuse* when $f(\mathbf{x}_i) > \tau$, where τ is a threshold parameter. In practice, log probabilities are used for numerical stability.

We test two methods for estimating the probabilities $p(t|y)$ from data. The first method, which we refer to as the *term prior*, defines for the i-th example:

$$p(t_j|y) = \frac{1 + t_{jy}}{n + \sum_{k=1}^{i} t_{ky}}$$

For the second method, referred to here as the *document prior* method, define d_{jy} as the number of examples seen with label y and containing feature t_j. The conditional probability is then defined as:

$$p(t_j|y) = \frac{1 + d_{jy}}{2 + i}$$

Updating the model with either method simply requires updating a set of feature counts after the label y_i is revealed for example $\mathbf{x}_i$. Thus, classification and training updates may both be completed in $O(s)$ time, where s is the number of non-zero elements of $\mathbf{x}_i$.

4.2 Perceptron with Margins

Perceptron with Margins [12] is a noise-tolerant [10] variant of the classical Perceptron algorithm, which uses a simple online update rule based on the 0-1 loss function. This discriminative algorithm has given strong results in TREC email spam filtering evaluations [22]. Perceptron with Margins learns a separating hyperplane, stored as a weight vector $\mathbf{w} \in \mathbb{R}^n$, and classifies examples using a linear function:

$$f(\mathbf{x}_i) = < \mathbf{w}, \mathbf{x}_i >$$

In practice, an offset value is implicitly included in the weight vector as $\mathbf{w}_0$, and each example $\mathbf{x}_i$ has $\mathbf{x}_{i0} = 1$. We use this convention for Perceptron with Margins, and for Logistic Regression, below. Abuse is predicted when $f(\mathbf{x}_i) > \tau$.

Perceptron with Margins begins with $\mathbf{w} \leftarrow 0$, and proceeds as follows. On each time step, after the label y_i is revealed, the following update is performed iff $f(\mathbf{x}_i)y_i < \rho$:

$$\mathbf{w} \leftarrow \mathbf{w} + y_i \eta \mathbf{x_i}$$

Here, ρ is a margin parameter. Thus, Perceptron with Margins updates the hypothesis whenever a mistaken prediction was made, or an example was found to lie within distance ρ of the separating hyperplane. This process has been likened to an inexpensive approximation of the maximum margin principle used by SVMs [10]. The parameter η controls the step-size, or learning rate. Training with class-specific misclassification costs may be implemented by using separate η_+ and η_- values for positive and negative example updates. Classification and training are both performed in $O(s)$ time.

4.3 Logistic Regression

Logistic Regression is a discriminative classification method, optimizing a logistic (sigmoid) loss function. Logistic Regression was recently proposed for email spam filtering [8], and has given state of the art performance at TREC [4]. We test both classical Logistic Regression and Logistic Regression with 2-Norm Regularization.

Like Perceptron variants, Logistic Regression stores a linear hypothesis with a weight vector $\mathbf{w}$. The prediction function maps an input example to the probability that the example has a positive label, based on that example's distance from the separating hyperplane [17]. That is:

$$f(\mathbf{x}_i) = p(y_i = 1|\mathbf{x}_i) = \frac{1}{1 + e^{<\mathbf{w}, \mathbf{x}_i>}}$$

We predict *abuse* whenever $f(\mathbf{x}_i) > \tau$.

The update procedure for Logistic Regression uses an online gradient descent method. We start with $\mathbf{w} \leftarrow 0$ and update for each new example. Assuming that $y \in \{0, 1\}$ rather than $y \in \{-1, +1\}$, so that y may be treated as the true probability of class membership, the update for $\mathbf{w}$ is:

$$\mathbf{w} \leftarrow \mathbf{w} + \eta \mathbf{x}_i (y_i - f(\mathbf{x}_i))$$

Like the methods previously described, both classification and updates are performed in $O(s)$ time.

Regularized Logistic Regression Logistic Regression is often considered to be subject to overfitting in the presence of noise, which could be problematic with noisy user flag data. One common strategy for reducing overfitting is *regularization*, which penalizes model complexity as measured by the magnitude of $\mathbf{w}$ using the 2-norm [17]. The Logistic Regression online update may be modified to incorporate 2-norm regularization [17] as follows:

$$\mathbf{w} \leftarrow \mathbf{w} + \eta \mathbf{x}_i (y_i - f(\mathbf{x}_i)) - \eta \lambda \mathbf{w}$$

Here, $0 \leq \lambda$ is a regularization parameter, where higher values lead to more aggressive regularization. Note that this method increases the computation cost of each training update from $O(s)$ to $O(n)$, making it linear in the total number of observed non-zero features in the entire data set rather than only the number of non-zero features in an individual example.

4.4 Relaxed Online SVMs

SVMs have long given state of the art performance on high dimensional data, such as text classification, but computation cost scales poorly with the size of the training data. Relaxed Online SVM (ROSVM), a sliding-window SVM approximation algorithm, was recently proposed for online email spam filtering [21], and gave state of the art performance on several tasks at TREC [3]. ROSVMs learn a linear hypothesis stored as a weight vector $\mathbf{x}_i$; thus, the prediction function $f(\mathbf{x}_i) =< \mathbf{w}, \mathbf{x}_i > +b$, and *abuse* is predicted whenever $f(\mathbf{x}_i) > \tau$.

The standard soft-margin SVM optimization problem for i examples is to minimize [19]:

$$\frac{1}{2}||\mathbf{w}||^2 + C \sum_{j=1}^{i} \xi_j$$

Subject to the constraint that for $1 \leq j \leq m$, $y_j f(\mathbf{x_j}) \geq 1 - \xi_j$ where each ξ_j is a slack variable allowing error, and C is a cost parameter. When C is set to a lower value, more emphasis is made regularization at the cost of training error. Class-specific misclassification costs may be implemented by using separate C_+ and C_- for positive and negative training examples. ROSVMs make this approach tractable for streaming data by implementing a sliding window [21]. Optimization is only performed on examples within this sliding window, which we set to a size of 1000 for this paper. Classification is performed in $O(s)$, but updates require computation roughly quadratic in the size of the sliding window.

4.5 Alternatives

We also investigated the utility of ensemble methods combining the output of several different filters, which were found effective for email spam [14]. Our experiments with these methods showed no improvement over the best filter included in the ensemble, as the filters tended to make correlated errors. We also experimented with online methods of ignoring or correcting labels that our filters suspected to be noisy, but this did not improve results.

4.6 Feature Sets

Each of the filtering methods requires comment text to be represented by feature vectors in $\mathbb{R}^n$. Our initial tests conducted on separate tuning data (Section 5.2) showed that the Naive Bayes variants performed best with a binary word based feature space, where a word is defined as a contiguous substring of case-normalized non-whitespace characters. The discriminative methods performed best with a binary 4-*mer* feature space, composed of the set of all (possibly overlapping) substrings of length 4, drawn from the standard ASCII character set that appear within the comment text. The feature vectors are then normalized with the Euclidean norm. This feature set gave state of the art results for several filtering methods at TREC [4] and on the small set of blog comment spam developed by Mishne [16, 21]. In our experiments, we applied these optimal pairings of feature-sets and filters, accordingly. In addition to the either word-based or 4-*mer* features, we included distinct binary features based on `blog` title, and others on `userid`.

5 Experiments

In this section, we detail our experimental framework and report results using user flag data for evaluation, both for testing individual filters and for comparing global versus per-topic filtering.

5.1 Experimental Design

For each filtering method, we performed separate online filtering experiments for each topic and a global

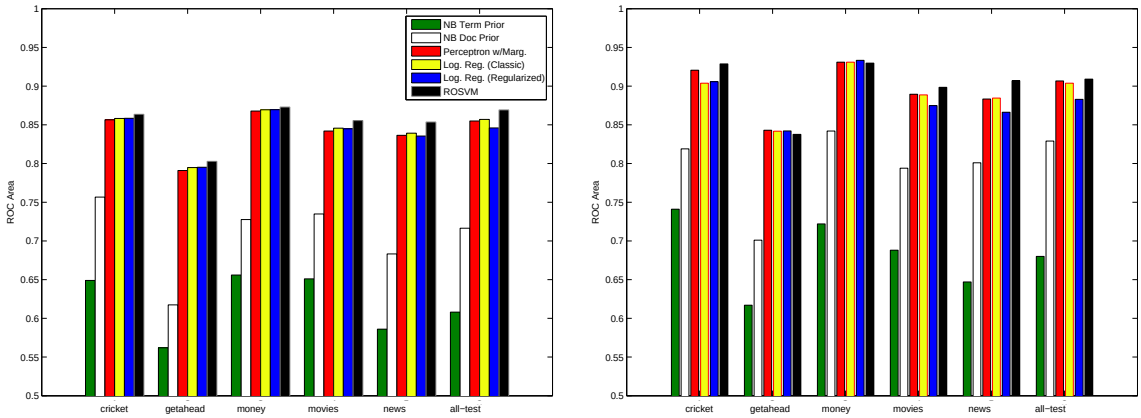

Figure 2: **ROCA results for User-Flag evaluation (left) and Gold Standard Evaluation (right).** Legend is the same for both graphs.

`all-test` experiment using data from all topics excluding `sports`. The `sports` topic was reserved for initial tests and parameter tuning. We used user-flag labels as ground truth for training, with flagged examples counting as positives.

Our primary evaluation measure is Area under the ROC Curve (ROCA), a standard measure which here may be statistically interpreted as the probability that a filter will correctly rank a randomly selected abuse comment as being more abusive than a randomly selected non-abuse comment. This is the standard evaluation measure for email spam filtering systems [6].

5.2 Parameter Tuning

Following standard machine learning methodology, we used the `sports` topic data as a separate tuning set to tune parameters for all filters before running the full experiments. Coarse grid search was performed to tune all parameters, using ROCA as our evaluation measure and user flag labels as ground truth for both training and evaluation.

The use of class-specific misclassification costs gave improved performance for each of the discriminative filters in tuning tests. This was due both to the fact that flagged comments are a minority of the data (see Table 1), and that the presence of a `flag` may be more reliable information than a `non-flag`. For Perceptron with Margins, tuning set $\eta_+ = 0.7$ and $\eta_- = 0.2$. For both Logistic Regression variants, $\eta_+ = 0.3$ and $\eta_- = 0.08$, and for 2-norm regularization $\lambda = 0.00004$. For ROSVM, $C_+ = 1.5$ and $C_- = 0.5$.

Note that the ROSVM parameter performed best with low values of the cost parameter C, which enforces regularization to help cope with label noise. This is a radical difference from email spam filtering evaluations, where very high values of C (representing minimal regularization) have been found most effective [7, 21] due to the lack of noise in the training labels.

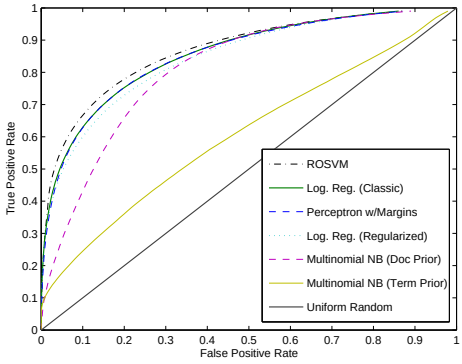

Figure 3: **ROC Curves using User Flag Evaluation, for `all-test`.**

5.3 Results Using User-Flags for Evaluation

ROCA scores using user-flag labels as ground truth for evaluation are given for each filter and task in the left graph of Figure 2. We make three observations from these results.

First, the discriminative classifiers strongly outperform the generative Naive Bayes classifiers. Within the Naive Bayes classifiers, the document prior was much superior to the term prior. ROSVM gives best results across the majority of tasks, but both Logistic Regression and Perceptron with Margins yield close performance and may be preferred for lower computational cost.

Second, using the 2-norm regularization for Logistic Regression was not beneficial, and actually reduced performance in almost all cases compared to the non-Regularized variant. We believe this is because in abuse filtering there are many rare-but-informative features, such as misspellings and intentional word obfuscations. The 2-norm regularization aggressively reduces the influence of rare features. The regularization provided by the low C values of ROSVMs appeared better to handle these sparse, informative features.

Third, the overall ROCA performance of the filters is far below the levels commonly seen in email spam filtering. As we show in Section 6, the noise in user flags causes this evaluation to be an under estimate of true performance.

5.4 Filtering Thresholds

The full ROC curves for the `all-test` task with all filters are given in Figure 3, showing the effect of varying the classification threshold between abuse and non-abuse. As discussed in the introduction, the goal of promoting civil discourse requires removing abusive comments from view; but to protect free speech we must be careful to keep false positive rates low. Thus, we would prefer the ROC curve to remain tight against the left vertical axis (showing 0 false positive rate) for as long as possible. The ROSVM, classical Logistic Regression, and Perceptron with Margins all perform well on this task, filtering out 30% of user-flagged abuse with a 1% false positive rate, and roughly half of all flagged about with a 5% false positive rate.

5.5 Global vs. Per-Topic Filtering

Looking at these initial results, it is natural to ask if it is better to apply one global filter for all topics or specialized filters for each topic. To answer this, we compared the results of each filter on the global `all-test` topic to the cumulative result across each of the per-topic tasks.

The results, shown in Table 3, show that topic specific filtering improves the results of the generative filters using Naive Bayes. This is because the distributions of abuse vary by topic; when all topics are combined the generative methods are less able to accurately estimate these distributions, reducing predictive performance. However, discriminative filters give nearly equivalent results for topic specific and global filtering, because these methods do not estimate distributions as an intermediate step. Thus, for the discriminative methods global filtering may be preferred for simplicity.

	Cumulative Topic Specific	Global All-Test
Naive Bayes Term Prior	0.630	0.608
Naive Bayes Doc. Prior	0.787	0.716
Perceptron w/Margins	0.848	0.855
Logistic Regression	0.852	0.857
Regularized Log. Reg.	0.850	0.845
ROSVM	0.862	0.863

Table 3: **ROCA results of topic-specific versus global filtering.** Generative methods benefit from topic-specific filtering, while discriminative methods are not significantly harmed by global filtering.

6 Gold-Standard Evaluation

To researchers familiar with email spam filtering, the initial results do not appear especially strong, despite the use of state of the art filtering methods. Is this because the problem of abuse filtering is inherently more difficult, due to problems of inter-annotators disagreement, or is it because the user flags are so noisy that they inherently under-estimate the true performance of the filters? To address this question, we constructed a *gold standard* set of examples for evaluation, taking effort to ensure that the labels for these examples were as trustworthy as possible. We then used this evaluation set to estimate the true performance of the filters, and to compare the effectiveness of automated filtering against user flagging.

6.1 Constructing a Gold Standard Set

We constructed a gold standard set using a methodology similar to that used by Cormack and Lynam to create gold standard labels for email spam [5], with efficient use of human adjudication effort.

We first sampled a pool of examples uniformly at random from each topic, stratifying the sampling by topic. This uniform sampling ensures that the gold standard set may be used for future evaluations. We then removed all examples from the pool for which the user-flag label and the predicted label given by each of four filters[3] unanimously agreed. Such unanimous labels were considered trustworthy [5] and were used as gold-standard labels for these examples.

From the sampled pool, there were 2,767 examples for which there was disagreement either among filter predictions or between a filter and the user flag label. These examples were human adjudicated. Three volunteer human adjudicators (one computer scientist and two non-computer scientists), who were each independently asked to label each example as `abuse`, `non-abuse` (ok), or `unsure`. Adjudicators were shown the text of the comment, the title of the blog in which it was posted, and the topic of the blog, along with brief guidelines for defining abuse. Adjudicators were not shown any of the predicted labels, nor the user flag label, nor any of the other adjudicators' ratings. Furthermore, the `userid` was withheld for privacy reasons.

Gold standard labels for the adjudicated set were determined by majority vote. The inter-annotator agreement for `abuse` and `non-abuse` was 0.742 ± 0.006, excluding `unsure` ratings, and the *kappa* measure of

[3]The four filters we used were multinomial Naive Bayes with the document prior, perceptron with margins, classical logistic regression, and ROSVM.

topic	Naive Bayes Term Prior	Naive Bayes Doc. Prior	Perceptron w/Margins	Logistic Regression	Regularized Log. Reg.	ROSVM	User Flags
cricket	0.384 ±0.075	0.323 ±0.071	0.677 ±0.080	0.616 ±0.081	0.626 ±0.081	0.717 ±0.078	0.735 ±0.073
getahead	0.132 ±0.059	0.191 ±0.069	0.500 ±0.097	0.441 ±0.095	0.456 ±0.095	0.485 ±0.097	0.753 ±0.086
money	0.313 ±0.062	0.398 ±0.067	0.711 ±0.069	0.711 ±0.069	0.695 ±0.070	0.727 ±0.068	0.791 ±0.062
movies	0.380 ±0.045	0.405 ±0.046	0.624 ±0.048	0.638 ±0.048	0.595 ±0.049	0.663 ±0.048	0.662 ±0.043
news	0.336 ±0.035	0.426 ±0.038	0.610 ±0.040	0.617 ±0.040	0.581 ±0.040	0.652 ±0.039	0.642 ±0.036
all-test	0.316 ±0.022	0.395 ±0.024	0.629 ±0.026	0.629 ±0.026	0.573 ±0.026	0.651 ±0.026	0.681 ±0.023

Table 4: **Results for F1 Measure, Gold Standard Evaluation.** F1 Measure is computed using precision and recall, where an abusive comment is considered a positive example. For all filters, the F1 measure was computed at the precision-recall break-even point.

	total sampled	fraction adjudicated	fraction corrected
cricket	994	0.22	0.07
getahead	1310	0.10	0.03
money	1275	0.22	0.05
movies	1909	0.34	0.12
news	2610	**0.43**	**0.16**
sports	1178	0.30	0.08
all-test	8096	0.30	0.10

Table 5: **Summary statistics for the gold standard evaluation set.** Adjudication and correction rates vary widely by topic. The `news` topic, in particular, required extensive adjudication of religious and racial comments.

agreement [2] was 0.48. Comments receiving predominantly `unsure` ratings – most often because the comment was in a non-English language – were given to a language expert for final adjudication. A small number of these were also labeled `unsure` by the language expert, and were removed from the set. The final gold standard set, which includes both adjudicated and unanimous examples, is described in Table 5.

6.2 Gold Standard Results

As before, each filter was tested on each task, using the online filtering methodology and the noisy user flag labels for training. Here, however, the evaluation was performed using gold standard labels, considering results only on examples in the gold standard set. The ROCA results for this evaluation are given in Figure 2 (right).

Note that the results for gold standard evaluation are uniformly higher than those using user-flag labels for evaluation. This shows that the user flag evaluation is, indeed, an under-estimate of true performance. However, it is interesting to observe that the *relative* performance of filters for tasks is largely preserved between both evaluations. Thus, in practical settings, it may be best to use the inexpensive user-flag labels for system tuning, testing, and monitoring to track relative changes. The more costly gold standard evaluation would only be required for occasional confirmation of true performance levels.

6.3 Filters vs. User Flags

The gold standard set also allows us to compare the performance of the filters with the process of user-flagging, itself. Using user-flags as a prediction, we computed *recall* (r), the fraction of all abuse that was detected, and *precision* (p), the rate of correct predictions found when abuse was predicted. For each task, we then computed the $F1$ measure [23], defined as $\frac{2pr}{p+r}$, for the user-flag predictions, and compared these to the $F1$ measure for each filter when the classification threshold τ was set at the precision-recall break-even point (where $p = r$).

The results, shown in Table 4 with 0.95 confidence intervals, show that user flags give slightly superior performance to the discriminative filters, and far outperformed the generative filters. An exception is the small `getahead` topic where lack of training data made the filters much inferior to user-flags. Furthermore, ROSVMs give improved performance on the contentious (Table 5) `news` task. However, in most cases, the confidence intervals for discriminative filters and user flags overlap, showing that these filters are able to approach the performance of user flagging. While it is encouraging that automated filters approach the effectiveness of user flags, as we discuss in Section 7 it is not necessary to choose between these approaches.

6.4 Filters vs. Dedicated Adjudicators

As noted before, the inter-annotator agreement for the human adjudicated examples was 0.742 ± 0.006. The discriminative classifiers approached this level of accuracy on the human adjudicated examples (a subset of the gold standard set). ROSVM and Perceptron with

Margins both gave accuracy of 0.728 ± 0.018 and Logistic Regression had accuracy 0.722 ± 0.018. For contrast, the user flag accuracy on the adjudicated examples was 0.689 ± 0.018. Thus, the discriminative filters agreed nearly as well with the human adjudications as the human adjudicators agreed with each other. This shows that these filters may be approaching the limits of human subjectivity on this task despite the noisy training labels.

7 Discussion

We have shown that effective filters for blog comment abuse may be trained using user flag labels, despite the noise inherent in this signal. Online SVM variants give best results, but other discriminative classifiers give nearly as strong performance and are considerably cheaper. Generative models, represented here by Naive Bayes, fare relatively poorly.

This filtering domain has proven to be considerably more difficult than the email spam filtering domain. This is because the labels for training are inherently noisy, there are more varied forms of abuse in this domain than in commercial email spam, and subtle cases of abuse and non-abuse are difficult to distinguish in this domain. Nevertheless, when plentiful training data is available, discriminative filters approach the performance of user flagging as a filtering methodology, and rival the effectiveness of dedicated human annotators. In the remainder of this section, we consider issues surrounding real world application of automated abuse filtering for blog comments, including plans for future work.

7.1 Two-Stage Filtering

In the real world application, it is not necessary to choose between automated filtering and user flagging. They can be used in series, with an automated filter serving as a first level filter, and user flagging serving as a second level. Initial offline analysis of this approach using the gold standard set shows that this approach can cut the amount of abuse shown to users for flagging down to a third of current rates with little loss of non-abuse comments. Although live testing will be required for confirmation, it is our belief that limiting the amount of abuse shown to users will increase their ability to flag effectively. This conjecture is supported by the observation that the topics with the lowest amount of abuse in our corpus (Table 1) also had the most accurate user flagging (Table 6).

7.2 Feedback to and from Users

Of course, any comments that are automatically filtered will not be seen by users, and thus cannot be used for training the filter. This *one-sided feedback scenario* can be harmful to certain online learning algorithms, causing them to filter over aggressively. However, it was recently shown that margin based learners such as Perceptron with Margins and online SVM variants can perform well in practice under this scenario [20]. Exploring this effect in blog comment abuse filtering is an area for future work.

A larger concern is the feedback that is provided *to* users, who will immediately see when their comments are filtered out. This will allow abusers to adopt a trial and error approach to posting abuse, which may be detrimental to the filtering performance. One possible alternative is to use a filter to adjust the *flagging threshold* needed to remove a comment from view. This way, comments predicted to be abusive would require fewer flags to be removed from view, and comments predicted to be non-abusive would require many more. This element of non-determinism would reduce abusers' ability to break the filter, while maintaining the ability of a community to enforce its standards.

7.3 Individual Thresholds

Part of the noise in user-flag data is caused by the inherent subjectivity of the flagging task: different users have different standards for acceptable modes of discourse. As we have seen, this makes it difficult to create a single filter with a single threshold for defining abuse or non-abuse. However, enabling individualized filters for unique users is costly and inefficient. An alternative would be to allow users to select their own *threshold* for defining abuse. Some users would prefer to see all comments, regardless of obscenities, while others would prefer to be shown only the most civil. This information could be efficiently stored as a user preference, and used at serving time to hide particular comments. Furthermore, such a mechanism would enable the collection of more fine-grained flagging information. Comments flagged by tolerant users are likely to be highly offensive, and may be scored differently than comments that are flagged by sensitive users but unflagged by others. In this way, we could tailor the needs of protecting civil discourse and free speech not only to a community, but to individual users as well.

8 Acknowledgments

We gratefully acknowledge Rediff.com for providing the corpus of blog comments and thank our volunteer adjudicators for their painstaking efforts.

References

[1] A. Bratko, B. Filipič, G. Cormack, T. Lynam, and B. Zupan. Spam filtering using statistical data compression models. *J. Mach. Learn. Res.*, 7:2673–2698, 2006.

[2] J. Carletta. Assessing agreement on classification tasks: the kappa statistic. *Comput. Linguist.*, 22(2), 1996.

[3] G. V. Cormack. TREC 2007 spam track overview. In *TREC 2007: Proceedings of the The Sixteenth Text REtrieval Conference*, 2007.

[4] G. V. Cormack. University of waterloo participlation in the TREC 2007 spam track. In *TREC 2007: Proceedings of the The Sixteenth Text REtrieval Conference*, 2007.

[5] G. V. Cormack and T. R. Lynam. Spam corpus creation for TREC. In *Proceedings of the Second Conference on Email and Anti-Spam (CEAS)*, 2005.

[6] G. V. Cormack and T. R. Lynam. Online supervised spam filter evaluation. *ACM Trans. Inf. Syst.*, 25(3), 2007.

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

[8] J. Goodman and W. Yin. Online discriminative spam filter training. In *Proceedings of the Third Conference on Email and Anti-Spam (CEAS)*, 2006.

[9] P. Graham. A plan for spam. 2002.

[10] R. Khardon and G. Wachman. Noise tolerant variants of the perceptron algorithm. *J. Mach. Learn. Res.*, 8, 2007.

[11] P. Kolari, T. Finin, and A. Joshi. SVMs for the blogosphere: Blog identification and splog detection. *AAAI Spring Symposium on Computational Approaches to Analyzing Weblogs*, 2006.

[12] W. Krauth and M. Mézard. Learning algorithms with optimal stability in neural networks. *Journal of Physics A*, 20(11), 1987.

[13] Y.-R. Lin, H. Sundaram, Y. Chi, J. Tatemura, and B. L. Tseng. Splog detection using self-similarity analysis on blog temporal dynamics. In *AIRWeb '07: Proceedings of the 3rd international workshop on Adversarial information retrieval on the web*, 2007.

[14] T. Lynam, G. Cormack, and D. Cheriton. On-line spam filter fusion. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 123–130, 2006.

[15] V. Metsis, I. Androutsopoulos, and G. Paliouras. Spam filtering with naive bayes – which naive bayes? *Third Conference on Email and Anti-Spam (CEAS)*, 2006.

[16] G. Mishne, D. Carmel, and R. Lempel. Blocking blog spam with language model disagreement. *Proceedings of the 1st International Workshop on Adversarial Information Retrieval on the Web (AIRWeb)*, 2005.

[17] T. M. Mitchell. Generative and discriminative classifiers: Naive bayes and logistic regression. In *Machine Learning*. http://www.cs.cmu.edu/∼tom/ mlbook/NBayesLogReg.pdf, 2005.

[18] A. Ng and M. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *Advances in Neural Information Processing Systems*, (14), 2002.

[19] B. Scholkopf and A. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.

[20] D. Sculley. Practical learning from one-sided feedback. In *KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2007.

[21] D. Sculley and G. Wachman. Relaxed online SVMs for spam filtering. In *The Thirtieth Annual ACM SIGIR Conference Proceedings*, 2007.

[22] D. Sculley, G. Wachman, and C. Brodley. Spam filtering using inexact string matching in explicit feature space with on-line linear classifiers. In *The Fifteenth Text REtrieval Conference (TREC 2006) Proceedings*, 2006.

[23] I. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques, 2nd ed.* Morgan Kaufman, 2005.