Learning to Perform Moderation in Online Forums

Andrew Arnt and Shlomo Zilberstein
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
{arnt,shlomo}@cs.umass.edu

Abstract

Online discussion forums are a valuable resource for people looking to find information, discuss ideas, and get advice on the Internet. Unfortunately, many forums have too much activity and information available, resulting in information overload. Moderation systems are implemented in some forums as a way to handle this problem, but due to sparsity issues, they are often not sufficient. In this paper we describe a novel method for learning from past moderations to develop a classifier that can perform automated moderation and thus address the sparsity problem. Additionally, we discuss the possibility of training a moderating classifier on a moderated forum and then applying it to an otherwise unmoderated forum.

1. Introduction

Online discussion forums provide a valuable resource for the exchange of information, insights, and ideas between people interested in a particular area. The distinguishing feature that forums have from traditional web sites is that users of the forum are able to add their own comments about the topic at hand. These comments are then displayed for other users to view and possibly reply to. This format allows for interaction and conversation between users of the forum.

Like much of today's Internet, online forums face the problem of information overload; if a forum becomes very popular and thus has many posts from many users, the sheer amount of information can be overwhelming. The reader is faced with the daunting task of sifting through hundreds of user comments, trying to find those that are the best.

Some forums try to combat these problems via *moderation*, where user comments can be given a score or deleted by a *moderator*. In forums where comments are given scores, adding to the score is called *positive* moderation, while lowering the score is *negative*. Users can then filter or sort comments according their aggregate scores. This can be thought as a kind of collaborative filtering [1] (also known as recommender systems [9]). In CF systems, people's preferences of products or topics are used to predict what products or topics a another person might be interested in. A relevant example is the GroupLens [8], a system for filtering Usenet newsfeeds. Users rate Usenet articles, and the GroupLens system recommends articles that other users with similar preferences have rated highly.

However, moderation systems still have some flaws. Much like the sparsity problem in CF systems[6], it is unlikely that moderation will be done to every single comment posted in a busy forum. Further aggravating the problem is that comments made in an older discussion rarely get moderated, as the majority of users and moderators have moved on to newer topics. This is not a problem for active discussions, but if a reader comes across an older topic (via a search engine, for example) the newest comments will generally be unmoderated.

CF researchers have addressed the sparsity problem by introducing elements of content-based filtering to their systems [6, 10]. Content-based filtering systems recommend items to users based only on similarity to the content of other items that the user has rated. A content-based filtering approach is taken by NewsWeeder [5], where a user rates netnews articles and is recommended other articles with similar content based on an MDL-based learning algorithm.

We propose that it is possible to learn how to perform machine moderation in a forum by finding patterns in the moderations made by humans. Consider this simple example: suppose a user in a forum consistently posts comments that are positively moderated by the human moderators. It is likely that other posts by that user will be of similar value, even if they have not been moderated as such.

We show how a machine moderator based on a naive Bayes classifier can be trained on a data set of previously moderated comments from a forum. This classifier is then used to predict with high accuracy what type of moderations are given to a set of test comments from the same forum. This can be thought of as improving the CF-like modera-

tion system by using a classifier trained on the content and features of rated comments.

The forum could then use the classifier to augment the existing moderation system. This has a few beneficial effects. First, the classifier can provide an initial moderation rating to new comments before any user moderation has taken place. Also, since many bad comments will already be moderated down by the classifier, human moderation can be focused on moderating up good comments (which intuitively seems to be a more difficult distinction for a learning algorithm to capture). Finally, the classifier can continue to update moderations even after the discussion is old enough that human moderations are very infrequent.

An additional problem with forum moderation is that it requires humans and thus adds a layer of complexity and expense that many forum managers cannot or choose not to handle. Because of this many forums are left unmoderated. We examine the feasibility of training a machine moderator in a moderated forum, and then using that classifier to moderate comments in a different, unmoderated forum.

In the next section we give background on the structure of forums and how moderation systems function. We describe the process of training the classifier in Section 2 and give results in Section 3. We discuss transferring a moderating classifier across forums in Section 4.

2. Experimental Design

2.1. Example forums

The first forum studied in this paper is Slashdot.org (SD),a popular website featuring science and technology news. After a story is posted on the site, readers of the site are encouraged to add their own comments on the story, which are appended to the bottom of the page for future readers to observe and possibly reply to. SD supports threading, allows anonymous posting, and does not have any deletion of user comments.

In the current SD moderation system, each logged-in user is occasionally and randomly issued a small number of "moderation points." With each point, the reader/moderator can take away or add a point to the moderation score of a single comment. The moderation score of a comment can range from -1 to 5.

In this work, the value of a comment is based on the final moderation score it achieves. Those comments with scores of -1 were classified as low value, those with scores between 0 and 2 were classified undetermined, and those with scores between 3 and 5 were labeled as high value. The label of undetermined reflects the fact that very little or perhaps mixed moderation has been done to the comment, while a score of 3 or more can only be achieved by positive moderation of the comment.

The SD moderation system suffers from a sparsity problem: At peak levels of moderation, when a story has just been posted for discussion, just 60 percent of comments posted will receive any kind of moderation. By the time the story is two hours old, under 20 percent of comments are being moderated. After seven hours. less than 10 percent receive moderation.

A smaller forum with similar subject matter is kuro5hin.org (K5), which was started in 1999. K5 has a moderation system that, while sharing the same goals as the SD system, differs from it in some important ways. In K5 moderation, comments users can moderate comments on a scale of 1-5. The score of a comment is then the mean of all ratings given to the comment. New, unmoderated comments have an initial score of "none". Furthermore, "trusted" users have the ability to rate a comment as 0, in which case only other trusted users will be able to view it. This means that obviously low value comments can be deleted in the eyes of regular users. For the experiments that follow, we do not have access to these 'deleted' comments. We consider comments with scores of 4 and above to be high valued, 2 or below low valued, with all other comments labeled as having undetermined value.

Although there are many types of forums on the web, in this paper we use SD and K5 as case studies. These forums are good candidates on which to perform automated moderation for a number of reasons. The high volume of user comments combined with the fact that low value comments are not deleted (in the SD case) means that there is an ample supply of both positive and negative instances of comment moderation available to use as training data. The threaded structure and user accounts contribute many possible attributes on which to train a classifier. The similar content matter of the two forums will be important when we examine the generalizability of moderating classifiers.

2.2. Data collection

To construct the data sets, SD and K5 topics and comments were downloaded. The SD data was all posted between July 1 and October 10, 2002, while the K5 data was from November 28, 2000 to February 9, 2003. Any hyperlinks in topic or comment text were retrieved and, when possible, converted into a plain text format, Topics/comments posted after September 20, 2002 (SD) or October 10, 2002 (K5) were withheld as testing data. There are 1918 topics with 466868 comments in the SD training set, with 535 topics and 121694 comments in the test set. The K5 data set contains 293623 comments under 2908 stories for training, with 126022 comments under 1008 stories for testing.

Table 1. Available attributes of comment c

Attribute	Description	Values
COMMENT AGE	How long topic was available for response before c was posted.	*numeric*
COMMENT LENGTH	Number of words appearing in the text of c.	*numeric*
BAD COMMENT PROB	The estimated probability that the text of c would receive negative moderation. A good indicator of profanity and misspellings.	*numeric*
NUM CHILDREN	The total number of comments posted in the thread rooted at c.	*numeric*
AVG CHILD VALUE	The average value of all comments posted under c.	*numeric*
MAX CHILD VALUE	The maximum value of all comments posted under c.	{high,low,undet}
PARENT VALUE	The value of the parent comment of c, if it has one.	{high,low,undet,noparent}
AUTHOR ID	The ID number of the author of c.	*numeric*
AUTHOR AVG VALUE	The average value of all comments in training set that author of c has posted.	*numeric*
AUTHOR NUM COMMENTS	The number of comments in training set posted by author of c.	*numeric*
TOPIC REL, TOPIC REL LINK	Estimated relevance of text of c to the text of parent topic, without and with including any linked text.	*numeric*
PARENT REL, PARENT REL LINK	Estimated relevance of text of c to parent comment text, without and with including any linked text	*numeric*

2.3. Attribute Extraction

Once the downloading was completed, attribute extraction was performed. Some of the attributes listed above are trivial to obtain, but we describe the procedure for extracting the more complex attributes here. All attributes were computed independently for each data set.

The BAD COMMENT PROB attribute was obtained by training a naive Bayes text classifier, such as the one presented in [7], on a set of documents consisting of all the high and low value comments in a training set, with the target values being '-' for low value comments and '+' for high value comments. Undetermined value comments were not used in this corpus due to the fact that many of there comments may be just unmoderated rather than some neutral target value and would thus hinder the discriminatory ability of the classifier. The trained classifier was then used to obtain a probability of '-' for each comment in the original training and test sets.

The AVG CHILD VALUE attribute is computed by giving a value of 1 to high value comments, -1 to low value, and 0 to undetermined. It is the mean score of all comments posted in the thread rooted at comment c.

For the TOPIC REL attributes, we first constructed a corpus C , where each document contained the text of a single topic T in a training set. Then for each comment c in that training set, a query was made consisting of the text of c and run against the corpus C using Inquery [2], a TF-IDF based retrieval engine. TOPIC REL is then the rank of c's parent topic T in the sorted list of all topics by estimated relevance returned by Inquery. TOPIC REL LINK was extracted by repeating the above procedure but including the text of any hyperlinked content in the documents (when the hyperlink appears in a topic) or queries (when the hyperlink is in a comment).

On the test set, a corpus C' was created, containing documents from the test set topics and enough of the newest training set topics so as to make C' contain as many documents as C. Corpus C' was then used to compute TOPIC REL and TOPIC REL LINK. The similar size of C and C' should result in attributes that are comparable over the training and test sets.

To determine the PARENT REL attributes, we used the following procedure. For each topic T in the training and test sets, a corpus C_T was created where each document in C_T contained the text of a single comment c posted under topic T. Then, for every comment c_i posted under T in response to some other comment c_p, we ran a query consisting of the text of c_i against corpus C_T. PARENT REL is then the rank of c_p in the ranked list, divided by the total number of comments in T. For those comments without parents, a negative constant was assigned for this attribute. PARENT REL LINK is obtained by the same procedure with hyperlinked text included in document and query text.

2.4. Training and Testing

The numeric attributes (as indicated in Table 1) in each training set were discretized using MDL method of [3]. A standard naive Bayes classifier was then trained using Weka [13], a suite of Java-based machine learning tools.

The classification of an unlabeled comment from a test set is fairly straightforward. Attributes are extracted from

Table 2. Test set confusion matrices for complete classifiers.

Slashdot

↓Act\Pred→	high	undet	low	% correct	% rev
high	5372	8708	492	36.87	3.38
undet	5749	87662	7765	86.64	
low	127	1507	4312	72.52	2.14

Kuro5hin

↓Act\Pred→	high	undet	low	% correct	% rev
high	18524	29188	806	38.18	1.66
undet	12006	55059	2552	79.01	
low	948	5518	1421	18.02	12.02

Table 3. Test set confusion matrices for high/low instances only classifiers.

Slashdot

↓Act\Pred→	high	undet	low	% correct	% rev
high	12964	13	1595	88.97	10.94
undet	60059	105	41012	0.10	
low	326	5	5615	94.43	5.48

Kuro5hin

↓Act\Pred→	high	undet	low	% correct	% rev
high	40897	0	7621	84.29	15.71
undet	47846	1	21770	0.14	
low	3749	0	4138	52.47	47.53

the comment, with numeric attributes discretized according to the intervals learned in training. The naive Bayes classifier then labels the instance as either high, low, or undetermined value. However, the attributes AVG CHILD VALUE, MAX CHILD VALUE, and PARENT VALUE depend on the value of other comments in the topic. For example, the value of a parent comment is unknown until that comment has been classified, and likewise the average score of a comment's descendants is also unknown. To address this problem, we first make one pass over all the comments in a topic, leaving any value-dependent features as unknown. Once initial classifications are made, we can compute these attributes using those classifications. We can repeat this process of classifying and updating score-dependent features until no classifications have changed from iteration to iteration or until some maximum number of iterations (in this case, 10) has been reached.

3. Experimental Results

The results on the test sets are presented in Table 2. In a confusion matrix [4], rows represent the actual class of instances, while the columns represent the classification predicted by the classifier. As we can see, the classifier tends to classify the great majority of comments as undetermined, regardless of the actual value. This is not surprising, given that in the SD case, 87 percent of the training comments were in the undetermined category (versus 9 percent high and 4 percent low), while in the K5 case, 56 percent were undetermined, with 39 percent high and 6 percent low.

The classifier is adept at classifying low valued comments in the SD case, classifying 73 percent correctly, with very few classified as high value. Classification of high valued comments, which intuitively seems to be the more difficult task is moderately successful, with 37 percent classified correctly and less than 500 classified as low value. The "% reversed" column lists the fraction of high valued comments classified low, and low classified high. When using a machine classifier in a forum, these are the events that we want most to avoid.

The K5 task proves to be more difficult, especially with respect to low value comments, where only 18 percent are classified correctly, with a 12 percent reversal rate. The most likely reason for this is that our dataset did not contain to the 'deleted' comments in the forum (as described in Section 2.1). These obviously low value comments tend to be the easiest to classify, due to their complete lack of relevance and content. Of course, if a forum manager decided to use an automatic moderator on their system, they would have access to such comments.

Note that if our focus is using the classifier to combat the issue of moderation sparsity, it is not desirable to train the classifier on a sparsely moderated data set. The classifier can be made more "aggressive" in assigning value by reducing the number of undetermined training cases. Table 3 contains results for classifiers trained on only the high and low value comments of a training set. We see that high valued instances are now classified correctly over 80 percent of the time in both forums. Low valued instances are classified with 94 percent accuracy in the SD task, while low valued comments remain challenging on the K5 task. Any instances classified as undetermined are a result of the smoothing done during training.

4. Generalizability

As discussed above, one of the motivations for this research is to examine the feasibility of using a classifier trained on one moderated forum to provide some rough moderation to a different, unmoderated forum with similar content and properties. This classifier would obviously not be able to use any author-derived attributes, as these cannot

Table 4. Test set confusion matrix for generalized high/low instances only classifier.

Slashdot (using K5 classifier)

↓Act\Pred→	high	undet	low	% correct	% rev
high	13225	0	1347	90.76	9.24
undet	82426	0	18750	0.00	
low	2803	0	3143	52.86	47.14

be transferred between distinct forums with distinct users. We call a classifier with no author features a generalized classifier.

However, the major problem is how to evaluate the performance of the classifier in the unmoderated forum. Since this new forum would have no user moderations with which to compare, we would have to rely solely on qualitative metrics, such as feedback from the users of the new forum.

To give some estimation of the generalizability of these classifiers, we trained a NB classifier withholding all author-based features on the K5 forum and evaluated its performance on our SD test set. Results are shown in Table 4. We see that this classifier gives acceptable performance, although the classifications are skewed toward high values.

Research is under way to improve cross-forum classification performance. We can obtain some author-based features in an unmoderated forum U by making an initial pass over a large number of topics and comments in U using a generalized classifier trained on a different, moderated forum M. Using the labels given by the M classifier, it is now possible to generate author-based features for future instances in U. Consequently, a full classifier from M can now be used to classify instances in U.

Additionally we can perform linear transformations on attributes in U, so that the distribution of values for each attribute is closer to the distribution seen in M. The best transformation can be learned by comparing the mean and variance of attributes in U to attributes in M.

5. Conclusions

We have shown that a naive Bayes classifier can be trained on a set of human-moderated comments in an online forum and then be used to predict, with high accuracy, the expected moderations of unlabeled comments in that same forum. This classifier can then be used to augment the existing human moderation scheme, addressing inherent sparsity problems. We also showed that it is feasible to train a classifier on a moderated forum, and then use that classifier in a different, unmoderated forum.

Future research includes developing an attribute to detect redundancy in comments: that is, a comment that supplies information already given earlier in the discussion. Also worth investigating is the possibility of detecting positive or negative tone in comments (as in [12]) as a basis for other features on which to train the classifier. Additionally, a feature that reflects the writing style and grammar usage of a comment may be valuable in some forums. A parser [11] could be used to estimate how well written individual sentences in a comment are.

Acknowledgments

This work was supported in part by the National Science Foundation under grants IIS-9907331 and INT-9612092. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of the sponsors.

References

[1] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proc. of UAI-98*, pages 43–52, 1998.

[2] J. P. Callan, W. B. Croft, and J. Broglio. TREC and Tipster experiments with InQuery. *Information Processing and Management*, 31(3):327–343, 1995.

[3] U. M. Fayyad and K. B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proc. of IJCAI-93*, pages 1022–1027, 1993.

[4] R. Kohavi and F. Provost. Glossary of terms. *Machine Learning*, 30(2/3):271–274, 1998.

[5] K. Lang. NewsWeeder: Learning to filter netnews. In *Proc. of the 12th ICML*, pages 331–339, 1995.

[6] P. Melville, R. J. Mooney, and R. Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *Proc. of AAAI-2002*, pages 187–192, 2002.

[7] T. M. Mitchell. *Machine Learning*. McGraw Hill, New York, NY, 1996.

[8] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. GroupLens: An open architecture for collaborative filtering of netnews. In *Proc. of ACM CSCW'94*, pages 175–186, 1994.

[9] P. Resnick and H. R. Varian. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997.

[10] B. M. Sarwar, J. A. Konstan, A. Borchers, J. Herlocker, B. Miller, and J. Riedl. Using filtering agents to improve prediction quality in the GroupLens research collaborative filtering system. In *Proc. of ACM CSCW'98*, pages 345–354, 1998.

[11] D. Sleator and D. Temperley. Parsing english with a link grammar. Technical Report CMU-CS-91-196, Carnegie Mellon University Computer Science, 1991.

[12] J. M. Wiebe. Learning subjective adjectives from corpora. In *Proc. of AAAI-2000*, pages 735–741, 2000.

[13] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools with Java Implementations*. Morgan Kaufmann, 1999.