Automatic Moderation of Comments in a Large On-line Journalistic Environment

Adriano Veloso adrianov@dcc.ufmg.br
Wagner Meira Jr. meira@dcc.ufmg.br
Tiago Macambira tmacam@dcc.ufmg.br
Dorgival Guedes dorgival@dcc.ufmg.br
Hélio Almeida helio@dcc.ufmg.br

Federal University of Minas Gerais, Belo Horizonte, Brazil

Abstract

On-line journalistic sites publish several news and stories every day. Readers of these sites may comment a story, and, as a consequence, a single story might receive thousands of comments. The quality of these comments may vary a lot, from spams to truly useful information. Separating good from bad comments is the primary goal of comment moderation. In this paper we address the problem of automatic moderation of comments in a large journalistic Web site. Participants of the site may interact with each other, constituting a large social network. We propose a classification technique which combines underlying implicit patterns in the comments' content with patterns hidden in the social network, and then uses the result for automatic moderation. We evaluate our proposed technique using a real collection of comments collected from the Slashdot forum. We compared the proposed technique against traditional ones, such as decision trees and SVMs. We observed that the proposed technique is very effective for scoring comments, reaching more than 96% of accuracy. Classifying comments seems to be a more complex task, and the proposed technique achieves almost 67% of accuracy. Further, the proposed technique is very fast, being able to classify and score hundreds of comments per minute.

Keywords

Text Categorization, Social Network Analysis, Filtering

1. Introduction

Moderation is the ability to give feedback on the quality of a given content. Several types of content may be moderated, such as articles, comments, wikis, blogs, and media files. Different methods of moderation may be applied, for instance, content may be classified (e.g., spam, informative etc.) or scored (e.g., rated in a -1−5 scale). Moderation is extremely important in mass communication systems, sorting good from bad content and helping readers to find useful information.

For instance, assume a large journalistic Web site, specifically the Slashdot forum[1]. Slashdot publishes several stories every day, and its readers may post comments to specific stories, and these comments must be analyzed and moderated.

More than a forum for publishing stories, Slashdot constitutes a large social network. Depending on the comments that were posted by a given user, she/he may acquire fans, friends, and enemies throughout her/his existence as a participant of Slashdot. This interaction among users may result in communities and groups of users who have similar opinions (i.e., friends and fans) or not (i.e., enemies).

Each published story may receive thousands of comments, resulting in a huge amount of content to be moderated. The immediate alternative is to increase the number of moderators. Still, if we consider the total time that is spent by all moderators, we reach an unwanted work of extraordinary magnitude. What is needed is an automatic technique, in order to reduce the time spent in moderating comments. In this paper we propose such a technique and evaluate it using a collection of 301,278 comments that were posted in response to 472 stories published in the Slashdot forum.

2. Automatic moderation of comments

Associative classification [3] is a technique that makes use of association rules to construct a classification model from the training data. In the following we define some key concepts, and then we introduce our associative classifiers.

Definition 1. [Comments] Let $\mathcal{C}$ denote the set of m comments $\{c_1, c_2, \ldots, c_m\}$, where each comment c_i is composed of a category k and a score s, along with a set of features (i.e., a feature set or feature vector). There are 7 different types of features (author, title, body, fans, foes, freaks, and friends).

Let $\mathcal{F}$ denote the set of all unique features. A feature set is simply a non-empty subset of $\mathcal{F}$, and it may contain different types of features. The support (or frequency) of the feature set $\mathcal{X}$ is the fraction of comments in the training data that contain $\mathcal{X}$ as a subset, given by: $\sigma(\mathcal{X}) = \frac{|\{c_i \in \mathcal{C} | \mathcal{X} \subseteq c_i\}|}{m}$. The feature set $\mathcal{X}$ is frequent if $\sigma(\mathcal{X}) \geq \sigma_{min}$, where σ_{min} is a user-specified minimum support threshold.

Definition 2. [Association Rules] The rule $\mathcal{X} \xrightarrow{\theta} k_i$ associates a feature set $\mathcal{X}$ to a category k_i. The support of the rule is given by $\sigma(\mathcal{X} \rightarrow k_i)$. The strength of the rule is given in terms of its confidence, defined as the conditional probability of the consequent when the antecedent is known, given by: $\theta = \frac{\sigma(\mathcal{X} \rightarrow k_i)}{\sigma(\mathcal{X})}$. The rule $\mathcal{X} \xrightarrow{\theta} k_i$ is strong if $\theta \geq \theta_{min}$, where θ_{min} is a user-specified minimum confidence threshold. There are several efficient algorithms for mining association rules [4].

The classification model is composed of strong rules. Each rule $\mathcal{X} \rightarrow k_i$ is considered as a vote of $\mathcal{X}$ for category k_i. Some votes weight more heavily than others, depending on the quality of the rule that is issuing the vote. The weight of each vote represents how many points the vote is worth, and it is calculated based on how frequent and strong the

[1] http://slashdot.org

corresponding rule is, given by: $\sigma(\mathcal{X} \rightarrow k_i) \times \theta(\mathcal{X} \rightarrow k_i)$. The weight is the ratio between the total number of points of each category and the total number of votes for this category. For example, category k_i, which receives n votes, will make $\frac{\sum_{j=1}^{n}(\sigma(\mathcal{X}_j \rightarrow k_i) \times \theta(\mathcal{X}_j \rightarrow k_i))}{n}$ points. The predicted category is the one that acquired the higher number of points.

An important concern is to select the rules to compose the model. There are two approaches for model induction: eager and lazy classifier. The *eager classifier* (EAC) induces a single model, and uses this model to classify all comments, that is, the model is induced before the comments that will be classified are known. In this case, the challenge for the classifier is in anticipating all the different directions in which it should attempt to generalize its training examples (i.e., what rules must be generated). In practice, the induced model is good on average, but fails in specific cases (i.e., comments with rare words or posted by not so frequent users). The *lazy classifier* (LAC) induces a specific model for each comment to be classified, that is, the induction process is delayed until a comment is given for classification. In this case, the classifier generalizes its training examples exactly as needed to cover the comment to be classified. The drawback is that the classifier has to induce several models, one for each comment to be classified, demanding more computational resources than its eager countpart.

3. Experimental evaluation

We divided our evaluation into two distinct tasks: scoring and classifying comments. Our proposed techniques are compared against SVMs [1] and C4.5 decision trees [2]. In all experiments we used 10-fold cross-validation and the final results of each experiment represent the average of ten runs. All experiments were performed on a Linux-based PC with a Intel Pentium III 1.5 GHz processor and 2.0 GBytes RAM.

3.1 Comment scoring

The first task is to estimate an integer value, the score of the comment. Figure 1(a) shows the accuracy achieved by the proposed approaches with different values of σ_{min}. As we can see, best accuracy is obtained when σ_{min} is between 0.05% and 0.1%. Lower values of σ_{min} results in a larger number of rules, but some of these rules may overfit the training data, hurting accuracy. On the other hand, higher values of σ_{min} results in fewer rules, increasing the chance of a scenario in which there is no rule to be applied to score a specific comment.

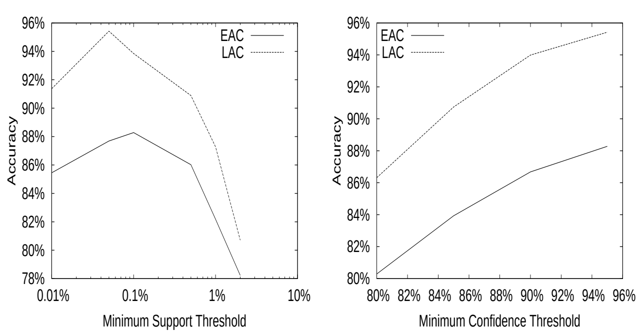

Fig. 1: *(a) Accuracy with σ_{min} (θ_{min}=95%), (b) Accuracy with θ_{min} (σ_{min}=0.05%).*

Figure 1(b) shows the accuracy achieved by the proposed approaches with different values of θ_{min}. As expected, best accuracy is obtained when higher values of θ_{min} are employed, since in this case more accurate rules are generated.

3.2 Comment classification

The second task is to predict the correct category of a given comment. This is a very hard task, because different moderators may have different opinions about similar comments. Table 1 shows a comparison of the different classification techniques. Similarly to the first task, LAC classifier is the best performer in terms of accuracy. C4.5 decision tree classifier is the faster one, but achieves just 40.82% of accuracy.

Technique	Accuracy	Exec. Time
LAC	66.30%	11,289 secs
EAC	45.23%	8,785 secs
SVM	59.09%	17,917 secs
C4.5	40.82%	8,219 secs

Table 1: *Accuracy and Execution Time (σ_{min}=0.1% and θ_{min}=95%).*

4. Conclusions and future work

In this paper we addressed the automatic moderation of comments in a social environment. We considered automatic moderation essentially as a classification problem, and we proposed approaches to address this task. The proposed techniques are based on associative classification. We evaluated two different associative classifiers, using a collection of comments posted to the Slashdot forum. The eager classifier builds a single classification model that is used to moderate all comments. The lazy classifier builds a specific classification model for each comment to be moderated. The lazy classifier shows to be more accurate and faster than a SVM classifier. As future work we will explore other application scenarios such as opinion retrieval, spam detection and sentiment analysis.

Acknowledgements

This research was sponsored by CNPq, Fapemig, Finep, and UOL (www.uol.com.br) through its *UOL Bolsa Pesquisa* program, processes 20060519184000a (Adriano Veloso and Wagner Meira Jr.) and 20060519234826a (Tiago Macambira and Dorgival Guedes).

References

[1] C. Chang and C. Lin. LIBSVM: A library for support vector machines. *Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm*, 2001.

[2] R. Kohavi, D. Sommerfield, and J. Dougherty. Data mining using MLC++: A machine learning library in C++. In *Tools with Artificial Intelligence*, pages 234–245, 1996.

[3] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In *Proc. of the 4th SIGKDD Conf.*, pages 80–86, 1998.

[4] M. Zaki, S. Parthasarathy, M. Ogihara, and W. Li. New algorithms for fast discovery of association rules. In *Proc. of the 3th SIGKDD Conf.*, pages 283–290, 1997.