Automated assessment of review quality using latent semantic analysis

Lakshmi Ramachandran  
Department of Computer Science  
North Carolina State University  
Raleigh, USA  
lramach@ncsu.edu

Edward F. Gehringer  
Department of Computer Science  
North Carolina State University  
Raleigh, USA  
efg@ncsu.edu

Abstract—Reviews are used not only in education to assess a student’s performance, but also in industry, to review code and other artifacts. It is therefore important to provide formative feedback to people making such assessment decisions and provide them with a cogent basis for making them.  
Quality of a review can be identified by performing a metareview. Metareviewing involves reviewing a review, in order to assess the quality of reviewing. Manual metareviewing can be tedious and cumbersome, and often results in poor-quality feedback because reviewers have not received adequate training in what constitutes a good review. Automated metareviewing can provide quick and reliable feedback to reviewers on their assessments.  
Quantifiable factors that help identify the quality of a review include quality and tone of comments, and the number of tokens each contains. We use machine learning techniques such as latent semantic analysis (LSA) and cosine similarity to classify comments based on their quality and tone. Our paper details experiments that were conducted on student review and metareview data by using different data pre-processing steps. We compare these pre-processing steps and show that when applied to student review data, they improve data quality and hence provide better classification of text. Our technique helps predict metareview scores for student reviews. We also present a proof-of-concept prototype built using Ruby and R.  

Keywords—quality of reviews; automated metareviewing; text mining; latent semantic analysis;

I. INTRODUCTION

Several factors help determine the quality of formative feedback. Quality of feedback depends on whether the reviewer was successful in identifying a problem in the author’s submission and whether he has provided suggestions or pointers to fix this problem. A review is considered to be of a good quality if it can help the author identify mistakes and learn possible ways of fixing them. Feedback is evaluated by a process referred to as metareviewing. Metareviewing involves carrying out a review of a review. Performing metareviewing manually can be a tedious and cumbersome process. Feedback quality can be poor, because of lack of training in review skills - the same problem that makes metareviewing necessary. Automating metareviewing, on the other hand, could provide quick and reliable feedback to reviewers on their assessments of authors’ submissions. Timely feedback on reviews could help metareviewers correct their assessments and provide more useful and effective feedback to authors. In this paper we automate metareviews by identifying the quality, tone and quantity of review comments.

Classifying textual review data can be a challenging task because of its highly unstructured nature. We therefore apply some pre-processing techniques to the review data. However, not all pre-processing techniques applied to data help in improving its quality [1]. In this paper we apply a set of pre-processing techniques (discussed in Section II) to the review data and perform experiments to determine the extent to which each of them improves the quality of data that is to be classified. We have shown that these pre-processing techniques perform better than the un-pre-processed data set.

We use latent semantic analysis (LSA) [2] to obtain a succinct representation of the review comments and then apply cosine similarity measure to identify the quality and tone of review comments. Review vectors are then formed using quality, tone and quantity of comments. In order to predict metareview scores, the vector for a new review is compared to vectors for previous human metareviewed reviews. The closest review’s scores and comments are predicted as the new review’s metareview scores and comments.

The rest of this paper is organized as follows. Section II discusses details of each of the pre-processing steps. Section III contains details of the technique used. Section IV discusses experiments that were conducted. Section V discusses the prototype that was constructed and Section VI concludes.

II. DATA PRE-PROCESSING

This section discusses some data pre-processing techniques, which are useful in identifying the quality and tone of review comments. Each of these pre-processing techniques is evaluated in Section IV to determine their impact on correctly classifying review comments.

A. Augmenting feedback with syntax information

Syntax information could prove to be useful in identifying well-written and coherent sentences, which can be readily understood by the author. Kaneijiya et al [1] found out LSA performed as well as an intermediate-level human evaluator but not as well as an expert-level evaluator. The authors...
note that since LSA uses a “bag-of-words” approach, it does not consider word order. For instance, we consider two comments “John should discuss the advantages of static analysis.” and “John analysis static.” Both of these comments receive the same similarity value when compared to the comment, “The author has discussed some important static analysis techniques, due to the presence of the words “static” and “analysis.” However, the first comment can be made sense of, while the latter may not be of much help to the author.

Thus, LSA must be augmented with syntactic information in order to improve its performance. In our approach this is achieved by tagging comments with syntax information such as part-of-speech tags.

B. Segmenting comments

Consider the following example, “The explanation was not sufficient but the submission had many good figures.” This example contains both praise as well as criticism. Hence, pre-processing should segment up comments at transition keywords or phrases, which helps accurately classify every section of the review comment. Transition keywords or phrases are those which modify the meaning of the part of the sentence that follows them and convey a meaning different from that of the preceding section. Segmenting comments would be useful in predicting its quality, as per Table II.

C. Elimination of negated adjectives

This pre-processing step involves replacing the negation of adjectives of the form “not”+“adjective” by the antonym of the adjective. In case of other verbs, adverbs etc. whose exact antonyms cannot be determined, we replace “not”+“word” by “un-”word. We do this to distinguish between the positive and negative usage of the same word. It helps distinguish utilization of the word in different contexts. For instance, if we have a comment of the form, “The author’s explanation was not bad,” the comment could be falsely classified as a criticism because it contains the word bad. This replacement is only done to distance the comment from those that appear to be semantically similar but in fact mean the opposite.

D. Pruning review comments

During pruning only descriptive words such as adjectives, adverbs, verbs in their different forms are retained, and all other words are removed from the data. This is done to reduce the noise in the data and thereby the feature set of words before the classification step is applied. The type of words that were retained for the classification step include: superlative and comparative adjectives and adverbs, verbs in their different tenses, coordinating conjunctions and prepositions.

III. Technique

This section discusses our technique for identifying the quality and tone of review comments. We briefly describe the reason for selecting this technique and also list the metrics we use to carry out our assessments.

A. Latent semantic analysis and cosine similarity

Latent semantic analysis has been widely used for automating essay grading [2]. In LSA, each essay is represented as a vector, whose features or dimensions are its words. In our case a review comment represents a document, with terms in a comment forming features of a document. A term-document matrix $M$ displays the relationships between words and comments across all review comments. In order to determine the degree to which a word is present in each document in the corpus, LSA applies truncated singular value decomposition (SVD) to the matrix. SVD breaks down the matrix into three component matrices from which a term-document matrix of reduced dimensions $M'$ is reconstructed.

Cosine determines the degree of similarity between documents, based on their terms in a term-document matrix. Review comments with similar term vectors (columns of $M'$) are considered to be close to each other. If two documents $D_1$ and $D_2$ are represented by vectors then the cosine similarity between the two can be determined using equation 1, where $\Theta$ is the angle between the vectors. Review comments with exact same vectors have a cosine similarity of 1, while those that are completely dissimilar have a similarity value of 0.

$$similarity(D_1, D_2) = \cosine(\Theta) = \frac{D_1 \cdot D_2}{\|D_1\|\|D_2\|} \quad (1)$$

Cosine of the new review comment with each existing review comment is determined and the one that is most similar to the new review is identified. Its class label is then assigned to the new review. This technique can be thought of as being similar to $k$-nearest neighbor where $k = 1$ with a similarity measure being used instead of a distance measure (Euclidean).

The assessment metrics used to determine the performance of LSA and cosine on our review data include precision, recall and $f$-measure.

B. Identification of review features

Since review comments are textual in nature, identifying quantifiable features suitable to represent them is quite difficult. The features we selected to represent reviews are based on metareview rubrics that are more commonly used by Expertiza [3].

---

1 Due to lack of space in this paper a copy of the metareview rubric has been placed at http://www4.ncsu.edu/~iramach/Images/metareviewRubric.jpg.
Table I

Table contains examples of different types of comments and their corresponding content type.

<table>
<thead>
<tr>
<th>Comment</th>
<th>Type of content</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Yes, flow between topics seems good.&quot;</td>
<td>praise</td>
</tr>
<tr>
<td>&quot;Well they could have done a little better in this case.&quot;</td>
<td>criticism</td>
</tr>
<tr>
<td>&quot;Yes the page talks about the topic but there is repetition of information which could have been avoided.&quot;</td>
<td>problem detection</td>
</tr>
<tr>
<td>&quot;Yes, But certain core concepts could have been explained more properly. For example testing has not been explained at all.&quot;</td>
<td>solution suggestion</td>
</tr>
</tbody>
</table>

Table II

Table shows how quality is determined based on the types of content a comment contains. Numbers inside the brackets indicate the number of instances (comment segments) of a content type that are required for a certain quality value.

<table>
<thead>
<tr>
<th>Praise</th>
<th>Criticism</th>
<th>Problem Detection</th>
<th>Solution Suggestion</th>
<th>Quality</th>
</tr>
</thead>
<tbody>
<tr>
<td>no/yes</td>
<td>no/yes</td>
<td>no</td>
<td>no</td>
<td>low</td>
</tr>
<tr>
<td>no/yes</td>
<td>no/yes</td>
<td>no</td>
<td>yes(≥ 2)</td>
<td>high</td>
</tr>
<tr>
<td>no/yes</td>
<td>no/yes</td>
<td>yes(≥ 2)</td>
<td>no</td>
<td>medium</td>
</tr>
<tr>
<td>no/yes</td>
<td>no/yes</td>
<td>yes(== 1)</td>
<td>no</td>
<td>high</td>
</tr>
</tbody>
</table>

1) Quality of comments: Since a single comment is segmented into smaller units for easier classification; we determine the classes (praise, criticism, problem detection or solution suggestion) [4] to which different parts of a comment belong. As the name suggests praise and criticism provide a piece of praise or criticism respectively, without containing any additional information. Problem detection comments help locate the source of the problem, while solution suggestion comments offer some additional information that the reviewer could use to improve his work. Based on the type of content a metareviewer would expect from a good review comment, we classify the quality of comments as high, medium and low. If comments are found to contain only criticisms but no problem detection information, then the review is considered to be of a low quality. In the case of problem detection, the reviewer identifies a problem “repetition of information” in the work and in the case of solution suggestion, the reviewer provides an example topic “testing” that the author could have included in his work (Figure 1).

2) Tone of comments: A review comment can have a positive, negative or neutral tone [5]. Opinion words of the type adjectives, adverbs help us determine the semantic orientation of a comment [6]. Based on the presence of positive descriptors such as such as “excellent”, “great” or negative descriptors such as “poor”, “copied”, reviews are classified with a positive or a negative tone.

Positive toned reviews are given a score of +1 and those with a negative tone a −1. Similar to identifying quality based on content type of the different reviews (Table II), tones across the different review comments or segments are summed and if the cumulative score is > 0, the review comment is said to have a positive tone, while a score < 0 results in a negative tone and a score of 0 gives the review comment a neutral tone. Examples of comments containing positive, negative or neutral tone are available in Table III.

3) Number of tokens: Number of tokens is associated with review quality, in that a good review must give the author sufficient feedback. For instance, consider a review comment of the form “I think the explanation of the Java evaluation options could be better organized.” The number of tokens is determined out to be 9, where the tokens considered include “better”, “could”, “evaluation”, “explanation”, “Java”, “options”, “organized”, “the” and “think”.

While quality of comments depends on the content type of comments provided by the reviewer, the tone depends on the use of positive or negative descriptors. Number of tokens measures the amount of feedback provided by the reviewer. These three features taken together help judge the quality of feedback provided by reviewers.

C. Predicting metareview scores

Review vectors are formed for each new review comment using each of the metrics listed above quality, tone and number of tokens. Similarly, we can generate vectors for reviews that have been metareviewed by human metareviewers. We compare the new review vectors to review vectors of human-metareviewed reviews and identify the metareviewed vectors closest to the new reviews. Scores of the closest metareview vector are set as those of the new review.

Metareview scores are predicted for each of four questions on a metareview rubric. These values can be used by a reviewer to quickly identify where his review is lacking. Responses to these questions in the form of predicted metareview scores could be used by reviewers to improve their reviews. Metareview comments of the closest metareview can also be used to provide some text-based feedback to the reviewers.

IV. Experiment

In this section we perform an experiment to show the importance of the different pre-processing steps and the prediction of metareview scores.

A. Data and tools used

The data was collected using Expertiza, a web-based collaborative learning environment [3]. We extracted 4083 review comments (from 681 reviews to a 6-question rubric)
Figure 1. Figure displays an average of the precision, recall and $f$-measure values of classification of review comments based on quality and tone. Figure compares the precision, recall and $f$-measure values across different data sets and each bar represents the result obtained by a single data set. The last bar shows the results obtained when classes were randomly assigned to review comments.

from assignments completed using Expertiza.\textsuperscript{2} 909 of these comments (after removing duplicates and empty data) were used for training. Data was then manually classified based on its content and tone as per the information in Sections III-B1 and III-B2.

We used the statistical analysis tool R to perform computations and analysis on the peer feedback data. Latent semantic analysis and cosine were implemented using the \texttt{lsa} library available with R. \texttt{tm} and \texttt{openNLP} were used to perform the text-mining and natural language processing related functionalities respectively. \texttt{Wordnet} was used to identify antonyms for the negated adjectives.

\textbf{B. Effectiveness of pre-processed data in classifying comments}

Our technique performs classification to identify the quality (based on content) of review comments and to determine their tone. We carry out our classification technique on the five data sets mentioned in section II.

Figure 1 shows us that precision, recall and $f$-measure values of all pre-processed data sets are better than that of the un-pre-processed set. High precision indicates that of the set of \textit{classified} comments, a large number was correctly classified. A high recall indicates that of the total set of \textit{actual} comments belonging to a class, a large number was correctly classified. $f$-measure provides a single measure of the technique’s performance by taking a harmonic mean of precision and recall.

In the case of data set which contains antonyms of negated adjectives, the precision (55.95\%) is same as that of the un-pre-processed data. This is likely due to the fact that only 46 (1.5\%) of the original data set contained negated adjectives. A significant finding from this study is that we observe an improvement, albeit small, in precision, recall and $f$-measure values when using syntax-augmented data. This shows that syntactically enhanced review comments could be successfully classified based on their semantic similarity with other comments. Kanejiya et al [1], who carried out their analysis on syntactically augmented student essay data, found out that while grouping data, syntactic similarity between them was given higher priority when compared to semantic similarity. They found that the essays that were placed closer to each other were not as close semantically as they were syntactically.

The difference between precision, recall and $f$-measure values between the un-pre-processed data set and the data sets containing segmented comments and antonyms of negated adjectives are not very high (Figure 1), because the number of comments that are segmented or contain antonyms of negated adjectives is very small as shown in Table IV. In the case of the pruned data set, we found that 660 (20.7\%) of the pruned comments (without descriptive words) were left empty. Since only 79.3\% of the pruned review comments were suitable for performing the classification, the impact of pre-processing is less than it might otherwise be. Our technique is likely to perform better for each of the different pre-processed data sets if it contained more review comments that can be segmented, or more comments that contained negated adjectives or fewer empty pruned comments. To see if our technique is truly adding value, we compare our results with those obtained by randomly assigning test comments to classes. Results are shown along the last bar on Figure 1. We can see that our technique produces high precision, recall and $f$-measure values with the different sets of pre-processed data.

\begin{table}[h]
\centering
\caption{Table contains examples of different types of comments and their corresponding tones.}
\begin{tabular}{|l|l|}
\hline
Comment & Tone \\
\hline
"Yes. The prose is easy to understand." & positive \\
"Somewhat. The beginning prose that introduces method missing has been rewritten, but I find it more confusing than before." & negative \\
"Author’s prose is easy to understand but too many quotations." & neutral \\
\hline
\end{tabular}
\end{table}

\begin{table}[h]
\centering
\caption{Table shows distribution of the number of comments that were segmented, contained negated adjectives and pruned completely across the entire set of test comments.}
\begin{tabular}{|l|l|}
\hline
Description & Values \\
\hline
segmented comments & 210 (6.6\%) \\
comments with replaced negated adjectives & 46 (1.5\%) \\
comments with empty comments after pruning & 660 (20.7\%) \\
\hline
\end{tabular}
\end{table}

\textsuperscript{2}Due to lack of space a copy of the review rubric has been placed at http://www4.ncsu.edu/~lramach/Images/ReviewRubric.jpg
Data sets

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Data sets</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Data without any pre-processing</td>
<td>55.01%</td>
</tr>
<tr>
<td>2</td>
<td>Data augmented with syntax information</td>
<td>55.16%</td>
</tr>
<tr>
<td>4</td>
<td>Segmented data</td>
<td>55.94%</td>
</tr>
<tr>
<td>5</td>
<td>Data with antonyms of negated adjectives</td>
<td>53.62%</td>
</tr>
<tr>
<td>6</td>
<td>Pruned Data</td>
<td>50.54%</td>
</tr>
</tbody>
</table>

C. Accuracy of metareview scores prediction

Metareviews are scored on a Likert scale of 1 to 5, where 1 is the lowest and 5 the highest. To predict metareview scores automatically, quality, tone and number of tokens of review comments are aggregated to form review vectors. We selected 724 previously (human) metareviewed reviews and split them into 400 for training and 324 for testing. Metareview scores for the test set were predicted by identifying the training review that was closest to each test review vector. Before determining the distance between vectors, review features were weighted based on how important the feature was in predicting metareview scores. Since quality is the most important feature we multiply it by 100. Tone is multiplied by 10 and we divide number of tokens by 10, since it is of less importance when compared to the other two features. The metareview scores of the closest training review are then predicted as the test review’s metareview scores. Table V shows the accuracy of prediction for each of the data sets.

Although ideally, this technique should perform well, that is, given two similar reviews their metareviews should also be the same, we find that the accuracy values are around 50%. This could be because these reviews were metareviewed by students who do not have adequate experience metareviewing others’ work. We also noticed that metareview scores tend to be fairly high, with most of the reviews getting a score of “5”. This shows that metareviewers tend to be easy on the reviewers by giving them high scores.

V. Prototype constructed using R and Ruby

We constructed a prototype (Figure V) using an R gem available for Ruby called RinRuby [7]. The tool accepts feedback in the form of comments and scores from a reviewer, and outputs information on the quality, tone and quantity of feedback provided. The reviewer could use this information to change his review and submit a better review.

VI. Conclusions

This paper uses effective data pre-processing techniques along with latent semantic analysis and cosine similarity to determine the quality, tone and quantity of review comments. This allows the quality of newly submitted reviews to be assessed automatically. We have shown that pre-processed data sets show an increase in the precision, recall and \( f \)-measure values of classification of the test review comments when compared to that of the un-pre-processed data set. We are working on integrating the automated metareviewing feature with Expertiza, which would help students get quick feedback on the quality of their reviews, and enable them to improve their reviewing, thereby helping other class members improve their work.

REFERENCES