Going beyond “Good Job”: Analyzing Helpful Feedback from the Student’s Perspective.

M Parvez Rashid, Yunkai Xiao, Edward F. Gehringer
Department of Computer Science
North Carolina State University
Raleigh, NC, USA
{mrashid4, yxiao28, efg}@ncsu.edu

ABSTRACT
Peer assessment can be a more effective pedagogical method when reviewers provide quality feedback. But what makes feedback helpful to reviewees? Other studies have identified quality feedback as focusing on detecting problems, providing suggestions, or pointing out where changes need to be made. However, it is important to seek students’ perspectives on what makes a review helpful to a reviewee. This study explores the helpfulness of feedback from students’ perspectives when the feedback contains suggestions or mentioned problems or both. We applied natural language processing techniques to identify suggestions and problems mentioned in peer reviews. We also analyzed important text features that are associated with suggestions or problems detected by the peer feedback. The result showed that students are likely to find a review helpful if a suggestion is provided along with the problem mentioned in the feedback rather than simply identifying the problem.

Keywords
Peer assessment, neural-network, natural language processing, correlation coefficient, suggestions

1. INTRODUCTION
Peer assessment has been proven to be an effective learning approach in both face-to-face and distance learning classes. It is especially useful in massive open online courses (MOOCs) where the potentially overwhelming number of students has no fixed bound. All of these students must be assessed by someone, and there are only a limited number of staff. Peer assessment can be as accurate instructor assessment, since artifacts are reviewed by multiple assessors who can invest more time than a teacher could [17]. It can also provide timely feedback [1] that helps students to focus on their weaknesses. Peer assessors pick up some of the feedback workload for instructors, who can then offer more help to students who are in need.

Peer assessors provide assessment in two forms. One is textual feedback, which takes the form of prose feedback to a peer. This is usually used as formative feedback. Another is numerical scores, on a Likert scale, which allows a summative grade to be calculated. Most online peer-assessment environments use both kinds of feedback. Studies have shown that students learn more from giving feedback than receiving it [13, 2, 7, 5] and giving feedback engage students in active learning [16]. It forces students to think metacognitively [4], and learn in-depth, as reviewing a peer requires a good hold on the topic [8].

However, the learning experience in a peer-review environment depends on the quality of reviews provided by the student peers. The goals of fairness and equity require that, insofar as possible, all students receive helpful formative feedback on their work. But, not all assessors provide constructive feedback, due to lack of knowledge in the topic or simply carelessness. To encourage and guide students in reviewing the artifacts, instructors typically need to scrutinize reviews manually. This consumes a good portion of the time that would be saved by having students provide quality feedback. An automated analysis could save considerable time. A few studies [20, 19] have tried to lessen the instructors’ assessment burden by automatically detecting characteristics of a quality review. That raises the question of what defines a quality review. According to Nelson and Schunn [10] high-quality feedback consists of (i) identifying a problem and (ii) suggesting a solution. However, their finding was based on students’ performance and not from their (students’) perspective. It is important to identify whether “quality feedback” is actually helpful to the reviewers, based on students’ opinion of what feedback is helpful.

In this paper, we propose a method using natural language processing (NLP) and neural networks to automate the process of analyzing and classifying reviews to discover whether they contain suggestions and/or problems. We analyzed the words that are used to include suggestions or problems detected by the peer feedback. Our goal is to answer the following research questions:

- RQ1: Can we build a model to accurately detect comments containing suggestions or detecting problems?
- RQ2: Are “quality comments”—those containing suggestions, detecting problems, or both—actually helpful from the student’s point of view?
2. RELATED WORK

This section discusses related work on identifying the properties of a "quality" or "helpful" peer assessment.

Nelson and Schunn [10] examined five features of feedback (summarization, specificity, explanations, scope, and affective language) that constitute good-quality reviews, and the correlations among them. Their study divided the features of feedback into cognitive and affective components. According to their findings, summarization, specificity, explanations, and scope are cognitive in nature. Cognitive features of a review are expected to most strongly affect understanding. This explanation helped us to identify suggestions and problem detection as a property of quality feedback.

An approach to improve review quality is to provide the reviewer with a rubric defining the characteristics of a quality review. Jaco du Toit [3] conducted a study to identify the impact of peer review on essay assignments. The study showed that giving students a rubric describing the characteristics of a good essay can provide them with the insight to produce better quality assessments than they would otherwise produce. But this study did not specify the qualities of a good review. When they received poor reviews, they were confused about the quality of their work, sometimes feeling a false sense of accomplishment. Rashid et al [15] analyzed rubric items to determine which of them induce peer reviewers to write quality feedback using NLP approaches. In their work quality feedback was identified if the review text contained a suggestion, detected a problem problem, or was localized (pinpointing the place where a revision should be made).

McGrath and Taylor did a study on students’ perception of helpful feedback for writing performance [9]. Their study defined quality feedback (“developed feedback”) as clear, specific, and explanatory in nature. They measured students’ perception of developed feedback by having them rate the feedback on a Likert scale. The results showed that students rated developed feedback highly for helpfulness.

A survey of 44 students done by Weaver showed that, in order to use the feedback, students needed advice (suggestions) [18]. The analysis of the feedback content and students’ responses uncovered that vague feedback (e.g., “Good job”) is unhelpful, lacking in guidance (void of suggestions), or focused on the negative (mentioning only problems), or was unrelated to assessment criteria.

Ramachandran et al. [14], developed an automated system to evaluate reviews and show how they compared to other reviews for the same assignment. They extracted attributes like relevance to the submission, content, coverage, tone, and volume of feedback to identify a good-quality review. They constructed word-order graphs to compare the reviews with submission text and extract features from the reviews.

To identify localization and make suggestions to improve the review Nguyen et al. [11] applied natural language processing techniques. They provided real-time formative feedback to reviewers on how to localize their review comments.

Zingle et al. [20] used neural-network approaches to find suggestions in the review text, and compared them against rule-based NLP approaches. In a similar work Xiao et al. [19] used NLP techniques with several ML and neural-network approaches to identify problem statements in review text. Our work takes this a step further and asks whether it is enough for a review to detect problems, or whether reviews that also make suggestions are more helpful.

3. DATA

Machine-learning and neural network-based models can perform as well or as badly as the data they are given. However, obtaining good labeled data is expensive. For the purpose of our experiment, we have collected labeled datasets for comments with three different characteristics:

- detects a problem: A review comment is labeled yes or no according to whether it detects a problem.
- contains a suggestion: A review comment is labeled yes or no according to whether it contains a suggestion.
- is helpful: A review comment is labeled yes or no depending on whether the reviewee found it helpful.

We acquired this labeled peer-review data from the Expertiza system in a systematic manner. Expertiza is a system to support different kinds of communications that are involved in the peer-assessment process. It supports double-blind communications between authors and reviewers, assessment of teammate contributions, and evaluations by course staff.

For the purpose of this study, we collected the data from Object-Oriented Design and Development course at NC State University for about three years. This course used the Expertiza system to manage the peer-review assessment process for evaluating the students. In each semester, this course typically assigns three peer-reviewed assignments to students, who work in teams consisting of two to four members. Even though the assignments are done in a group setting, the submissions are reviewed by individual students from other groups. After receiving the reviews from peers, teams revise their work and resubmit it for grading. The second round of the assessment is generally summative, where along with textual comments, the peer-reviewers assign scores to the submission.

Generally, a small number of people cannot annotate a large dataset. It is better to have a large number of people each undertake a small number of annotation tasks; this lessens the chance that an annotator will become fatigued and assign inaccurate labels. We engaged students in the labeling task by offering a small amount of extra credit. After receiving peer feedback, students were asked to label the feedback to identify whether the reviewer mentioned a problem or suggested a solution. They were also asked whether they considered the feedback to be helpful. In different assignments, students were asked to label the feedback for different characteristics; the same comments were not necessarily labeled for all three characteristics. After labeling was complete, the course instructor and TAs spot-checked the data.
Table 1: Sample review comment and annotations done by students ('1' indicates 'yes' and '0' indicates 'no')

<table>
<thead>
<tr>
<th>Review Comment</th>
<th>Detects Problem</th>
<th>Gives Suggestion</th>
<th>Is Helpful</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Travis CI Build is Failing as of now. No conflicts as per the GitHub report.</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes, the explanation is elaborate and complete.</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Since the build failed, I would not recommend adding it to the production server yet.</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Test Plan is too verbose. Trivial areas can be trimmed off.</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The team needs to look into Travis CI log &amp; 1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Many test cases in terms of controllers, but none for models.</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The build is failing due to 4 failures in the model specs.</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The writeup is clear.</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Since the build failed, I would not recommend adding it to the production server yet.</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 1: Flow diagram of peer review and feedback annotation process

that each student labeled. If any labels were found to be incorrect, the data labeled by that student was excluded from the dataset.

Since the reviews were done on team projects, and labeling was done individually, two to four students had the opportunity to label (or “tag”) the same review comments. If multiple students did tag the same comment, inter-rater reliability (IRR) could be calculated. We chose Krippendorff’s α \([6]\) as the metric for IRR. We chose this metric because it is not impacted by missing ratings, which were common since not all students availed themselves of the extra-credit opportunity. In an effort to use only the most reliable labeling, we included only labels that were assigned (or not assigned) by all the students in the team that was reviewed. This allowed us to raise Krippendorff’s α of our dataset from 0.696 to 1. Figure 1 shows the peer-review and annotation process.

Following the described process we accumulated 18,392 annotations for problem detection, 7,416 for suggestion-detection and 3,970 for helpfulness-detection datasets. All the three datasets have an equal ratio of the binary class labels (i.e., they are balanced). Sample comments from the three datasets are shown in Table 1.

4. METHOD

Our goal in this study is to analyze students’ perspectives on helpful comments that mentioned problems and/or suggestions. To conduct the study, we had students annotate comments on the basis of whether they found them helpful. We need an automated process to identify those review comments that contain suggestions and/or problem statements. We first train a model (the problem-detection model) to classify reviews that contain a problem statement by training and testing with the problem-detection dataset. We build a second model (the suggestion-detection model) to classify the presence of suggestions in a review comment by using the suggestion-detection dataset. As model performance matters, we applied several ML and neural-network models to pick the most accurate models for annotating the helpfulness-detection dataset. Figure 2 shows the annotation process of the helpfulness-dataset using the models.

When approaching a classification problem by any type of machine-learning (ML) or neural network models, there are many different approaches to choose from. No one model is best for all problems. In our study, we have chosen Support Vector Machine (SVM), Random forest (RF), classical ML models and compared their performance with Bi Directional Long Short-term Memory (Bi-LSTM), and Bidirectional Encoder Representations from Transformers (BERT) models. We used TF-IDF for ML models and Global Vectors for Word Representation (GloVe) for Bi-LSTM to perform word vectorization. Before we applied any word vectorization techniques, we cleaned the text by removing URLs, stop words, and applying stemming. We use our problem-detection dataset and suggestion-detection dataset on these model with 80:10:10 ratio for training, testing and validation.

4.1 Classical Machine-Learning Models

4.1.1 Input Embedding with TF-IDF

Machine-learning models are suitable for capturing complex relationships between the input data. But they require numeric input. The review data that we have in our dataset is textual. We have to convert them to numbers and also allow the model to capture the important features of the text. One way to do that is term frequency-inverse document frequency (TF-IDF). TF-IDF measures the importance of a word in a document using statistical calculation. If a word appears more times in a document the importance of the word in the document increases proportionally. We used
Figure 2: Annotation process of the helpfulness-dataset for mentioned problem and suggestions in the comments using models. The models were trained for detecting problems or suggestions mentioned in the review text. The training datasets were annotated by human (students).

scikit-learn [12] library to implement TF-IDF and vectorize the words in the feedback.

4.1.2 SVM
SVM is very popular for high accuracy and low computational cost. For a classification problem between two classes, SVM maximizes the margin of the separation plane between the two classes. We provided the feature vector of the reviews converted by TF-IDF to the SVM model to classify the review for having a particular property (contains problem or suggestion in the comment). We applied a grid search to find the best inverse regularization parameter $C$. 

4.1.3 RF
We used Random Forest for its popularity to make more accurate classification with a simple approach. RF makes an ensemble decision from a forest consisting of multiple uncorrelated decision trees. The general idea of the RF is that the decision from individual decision trees increase the accuracy of overall result. We varied the number of decision trees and depth of the trees to get the best result. We used TF-IDF for making feature vectors from the review text.

4.2 Neural-Network Models

4.2.1 Input Embedding
Neural network models are popular for text classification tasks. However, to improve the performance of the neural-network models on the text data, it is necessary to represent the data that is suitable for the model to work with, and without losing the underlying latent relations among the features of the data. For our experiment we have used GloVe with Bi-LSTM. GloVe not only measures the statistical significance of words, it also considers the statistical co-occurrence and semantic relation of the words.

4.2.2 Bi-LSTM
Bi-Long Short-Term Memory is in general used for sequential data classification tasks. It is a good fit for peer-review texts. Review comments are sequential data, and the words of the text have latent semantic and contextual relations with each other. As Bi-LSTM model takes input from both right and left direction of the text, it can capture the relationship between the words in texts occurring in any order.

<table>
<thead>
<tr>
<th>Table 2: Hyperparameters of Models</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
</tr>
<tr>
<td>SVM</td>
</tr>
<tr>
<td>RF</td>
</tr>
<tr>
<td>Bi-LSTM</td>
</tr>
<tr>
<td>BERT</td>
</tr>
</tbody>
</table>

4.2.3 BERT
BERT is based on Transformer model and use attention mechanism to learn the contextual relations of the words in a sentence. Being a bi-directional input reader, BERT learns the context of word in sentence by considering words occurring before and after.

5. RESULTS
In this study, if a feedback comment mentions problems and/or suggestions, we are considering it to be quality feedback. Our first step is to construct two separate models where one identifies whether feedback contains a problem statement and another identifies whether feedback contains a suggestion. To identify the best-performing models we trained and tested the performance of several classical ML models and neural-network models and compared their performance.

RQ1: Can we build a model to accurately detect comments containing suggestions or detecting problems?

Figure 3 reports the comparison of the F1-score values of the classical machine-learning (ML) models and neural-network models on the problem-detection dataset and suggestion-detection dataset. To compare the performance of the models we use the F1-score, as this represents the harmonic mean of precision and recall.

- On the problem-detection dataset: Among the classical
Figure 3: F1-score comparison to measure performance on classifying review text on problem detection and suggestion detection using classical ML and neural-network models. In overall F1-score comparison, the BERT model shows the best performance.

- **On the suggestion-detection dataset**: BERT achieved the highest F1-score, 0.91. Among the classical ML models, SVM achieved the highest F1-score, 0.87.

To gain a deeper insight into the words that are highly correlated with text where a problem mention or suggestion is mentioned, we analyzed the top 10 positive and negative correlation coefficient values calculated by the SVM model. Figure 4(a) shows the coefficient values that the problem-detection model has calculated for various words. Note that it has positive coefficient values for words such as “however”, “but”, and “not”. In the English language these words are more likely to be used when stating problems. Similarly words like “yes”, “completed” and “good” are not likely to occur in a problem statement. Figure 4(b) shows that the suggestion-detection model has positive coefficient values for words like “should”, “would”, “more”, “suggest”. These words are likely to be used in suggestions. On the other hand, words “yes”, “completed”, and “cannot” are more likely not to be used to express suggestions; thus they have negative coefficient values.

As BERT outperformed all other models on both problem and suggestion datasets, we trained two separate BERT models to annotate the feedback comments contained in the helpfulness-detection dataset. The BERT-created annotations recorded whether each comment in the helpfulness-detection dataset detected a problem or offered a suggestion. The models annotated each comment with either “1” or “0”, indicating having the property or not. We perform an **and**-operation using the BERT-created annotations. If both the problem and suggestion were mentioned in a comment the **and**-operation yields 1 otherwise 0. The resulting helpfulness-detection dataset is shown in Table 3.

RQ2: Are “quality comments”—those containing suggestions, detecting problems, or both—actually helpful from the stu-
The results indicate that the automated annotation per-
problem nor contained a suggestion. Conversely, of the comments that the students
helpful, 64% of them both mentioned a problem and gave a
helpful). Among the comments that the students considered
students to improve their work.

To summarize the Venn diagram analysis, comments that
students found helpful mostly detected problems and/or
mentioned in the feedback. Out of these 1,417 helpful
comments, 912 of them were machine-annotated as contain-
ing both problem detection and suggestions. A total of 568
helpful comments did not have any problem or suggestion
mentioned, based on machine-annotation.

On the other hand, out of the 1,985 comments that were
human-annotated as not helpful [Figure 5(d)], 673 comments
were annotated as having either a problem and/or sugges-
tion mentioned. Among those 673 comments, only 174 were
annotated as having both suggestion and problem mentioned.
A total of 1,312 comments that did not have any problem or suggestion
mentioned were annotated as not helpful by the
students.

To summarize the Venn diagram analysis, comments that
the students found helpful mostly detected problems and/or
contained suggestions. However, among those comments
noting suggestions and/or problems, students annotated as
helpful mostly comments that both pointed out problems
and gave suggestions. This indicates that peer feedback is
more helpful to the students when a suggestion is given in
a comment that detects a problem. On the flip side, Figure
5(b) suggests that students rarely find comments helpful
when they do not mention any problem or contain a sugges-
tion.

RQ3: Can an automated process effectively identify helpful
feedback?

A key question is whether comments automatically anno-
tated as “quality” (meaning that they both identified a prob-
lem and gave a suggestion) were the same comments that the
students considered helpful (that they manually labeled as
helpful). Among the comments that the students considered
helpful, 64% of them both mentioned a problem and gave a
suggestion. Conversely, of the comments that the students
labeled as not helpful, 66% of them neither mentioned a
problem nor contained a suggestion.

The results indicate that the automated annotation per-
7. REFERENCES


