Improving Peer Feedback Prediction: The Sentence Level is Right

Huy V. Nguyen
Department of Computer Science
University of Pittsburgh
Pittsburgh, PA 15260
hvn3@pitt.edu

Diane J. Litman
Department of Computer Science & LRDC
University of Pittsburgh
Pittsburgh, PA 15260
litman@cs.pitt.edu

Abstract

Recent research aims to automatically predict whether peer feedback is of high quality, e.g. suggests solutions to identified problems. While prior studies have focused on peer review of papers, similar issues arise when reviewing diagrams and other artifacts. In addition, previous studies have not carefully examined how the level of prediction granularity impacts both accuracy and educational utility. In this paper we develop models for predicting the quality of peer feedback regarding argument diagrams. We propose to perform prediction at the sentence level, even though the educational task is to label feedback at a multi-sentential comment level. We first introduce a corpus annotated at a sentence level granularity, then build comment prediction models using this corpus. Our results show that aggregating sentence prediction outputs to label comments not only outperforms approaches that directly train on comment annotations, but also provides useful information for enhancing peer review systems with new functionality.

1 Introduction

Peer review systems are increasingly being used to facilitate the teaching and assessment of student writing. Peer feedback can complement and even be as useful as teacher feedback; students can also benefit by producing peer feedback. Past research has shown that feedback implementation is significantly correlated to the presence of desirable feedback features such as the description of solutions to problems (Nelson and Schunn, 2009). Since it would be very time-consuming for instructors to identify feedback of low quality post-hoc, recent research has used natural language processing (NLP) to automatically predict whether peer feedback contains useful content for guiding student revision (Cho, 2008; Ramachandran and Gehringer, 2011; Xiong et al., 2012). Such real-time predictions have in turn been used to enhance existing online peer-review systems, e.g. by triggering tutoring that is designed to improve feedback quality (Nguyen et al., June 2014).

While most prior research of peer review quality has focused on feedback regarding papers, similar issues arise when reviewing other types of artifacts such as program code, graphical diagrams, etc. (Nguyen and Litman, July 2013). In addition, previous studies have not carefully examined how the level of prediction granularity (e.g. multi-sentential review comments versus sentences) impacts both the accuracy and the educational utility of the predictive models. For example, while the tutoring intervention of (Nguyen et al., June 2014) highlighted low versus high quality feedback comments, such a prediction granularity could not support the highlighting of specific text spans that also might have been instructionally useful.

In this paper, we first address the problem of predicting **feedback type** (i.e. *problem, solution, non-criticism*) in peer reviews of student argument diagrams. In *problem* feedback, the reviewer describes what is wrong or needs to be improved in the diagram. In *solution* feedback, the reviewer provides a way to fix a problem or to improve the diagram quality. Feedback is *non-criticism* when it is neither a *problem* nor a *solution* (e.g. when it provides only positive feedback or summarizes). Examples are shown in Figure 1.[1]

The second goal of our research is to design our prediction framework so that it can support real-time tutoring about feedback quality. We hypoth-

[1]Our peer review corpus comes from a system that uses an end-comment feedback approach as shown in Figure 1. While it is possible to instead directly annotate a reviewed artifact, this has been shown to encourage feedback on low-level issues, and is not good for more global feedback.

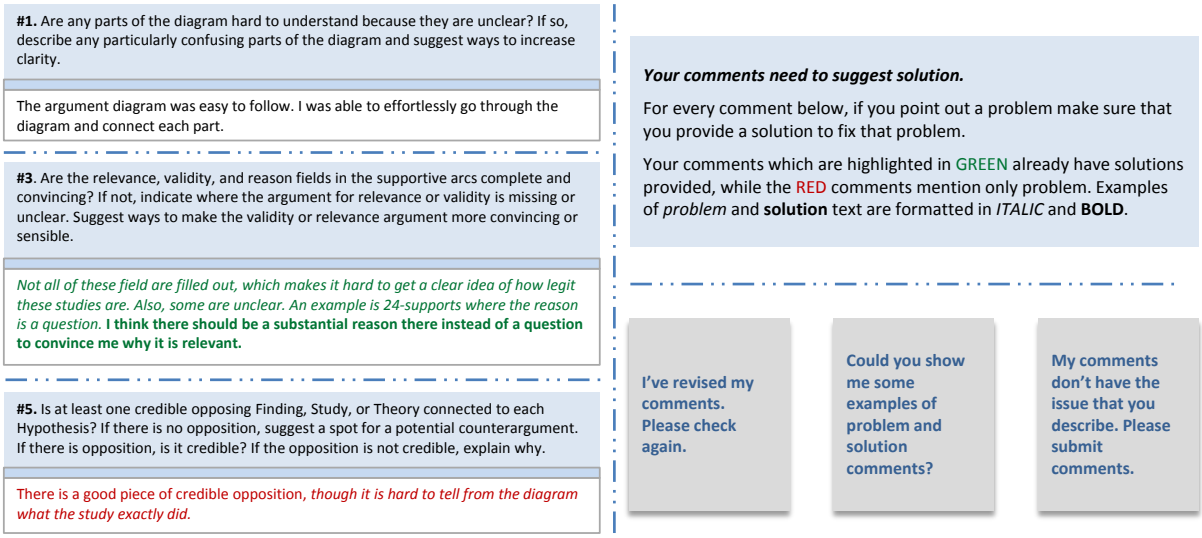

Figure 1: A mock-up interface of a peer review system where the prediction of feedback type triggers a system tutoring intervention. Left: three sample feedback comments including a *non-criticism* (top), a *solution* (middle), and a *problem* (bottom). Right-top: a system tutoring intervention to teach the student reviewer to provide a solution whenever a problem is mentioned. Right-bottom: possible student responses to the system's tutoring.

esize that using a student's own high-quality reviews during tutoring, and identifying the explicit text that makes the review high quality, will help students learn how to improve their lower quality reviews. To facilitate this goal, we develop prediction models that work at the sentence level of granularity.

Figure 1 presents a mock-up of our envisioned peer review interface. To tutor the student about solutions (figure right), the system uses live examples taken from the student's current review (figure left). Color is used to display the feedback type predictions: here a *non-criticism* is displayed in black, while the criticisms that are positive and negative examples of *solution* are displayed in green and red, respectively. In addition, to help the student focus on the important aspect of the (green) positive example, the sentence that actually specifies the solution is highlighted in bold.

This paper presents our first results towards realizing this vision. The contributions of our work are two-fold. First, we develop a sentence-level model for predicting feedback type in a diagram review corpus. While our peer review system works at the level of *feedback comments* (text of each box in Figure 1), we find it is more accurate to annotate and predict at finer-grained granularity levels, then use these predictions to infer the comment's feedback type. By introducing a small overhead to annotate peer feedback, we created a phrase level-annotated corpus of argument diagram reviews. Our experimental results show that our learned prediction models using labeled sentences outperform models trained and tested at comment level. In addition, our models outperform models previously developed for paper rather than diagram feedback, and also show potential generality by avoiding the use of domain-specific features. Second, we demonstrate that our sentence-level prediction can be used to support visualizations useful for tutoring. Particular sentences that are predicted to express the comment's feedback type are highlighted for instructional purposes (e.g. the bold highlighting in Figure 1).

2 Related work

In instructional science, research has been conducted to understand what makes peer feedback helpful. At the secondary school level, Gielen *et al.* (2010) found that the presence of justification in feedback significantly improved students' writing performance. At the university level, Nelson and Schunn (2009) found that feedback on papers was more likely to be implemented when the feedback contained solutions or pinpointed problem locations. Lippman *et al.* (2012) found that similar feedback properties led to greater implementation

of feedback on diagrams as well.

Building on such findings, researchers have begun to develop automated methods to identify helpful feedback. Cho (2008) was the first to take a machine learning approach. Peer feedback, i.e. comments, were manually segmented into idea units[2] and human-coded for various features including problem detection, solution suggestion, praise, criticism, and summary. Feedback was then labeled as helpful or not-helpful based on the presence of such features. The study showed that feedback could be classified regarding helpfulness with up to 67% accuracy using simple NLP techniques including ngrams and part-of-speech. Our work is different from (Cho, 2008) in that we focus on predicting particular feedback types (i.e. solution and problem) rather than helpfulness in general. Also, as the raw feedback to peer-review systems is typically at the comment-level, and being aware that idea-units are difficult to automatically segment, we instead predict at the sentence-level to make model deployment more practical.

Our work is more similar to (Xiong and Litman, 2010; Xiong et al., June 2010; Xiong et al., 2012), in which NLP and machine learning were used to automatically predict whether peer reviews of student papers contained specific desirable feedback features. Xiong and Litman used NLP-based features including paper ngrams, predefined keyword lists, and dependency parses to predict feedback type. For feedback of type criticism, they also developed models to further predict problem localization and solution. Following (Cho, 2008), Xiong and Litman evaluated their models on peer review data that had been manually segmented into idea units. As noted above, the difficulty of automatically segmenting raw comments into idea units makes deployment of such models less practical than our sentence-level approach. Also like Cho (2008), while their models predicted a label for each idea unit, the relevant text that led to the prediction was not identified. We will address this limitation by introducing a more fine-grained annotated corpus.

Regarding peer reviews of student argument diagrams rather than papers, Nguyen and Litman (July 2013) developed a rule-based algorithm for predicting feedback that contained localization text (e.g. "Hypothesis 4"). Their approach was to first identify common words between a peer comment and its diagram, then classify phrases containing these words into different localization patterns. Although we similarly focus on diagram rather than paper feedback, our work addresses a different prediction task (namely, predicting feedback type rather than localization). We also use statistical machine learning rather than a rule-based approach, in conjunction with more general linguistic features, to allow us to ultimately use our models for papers as well as diagrams with minimal modification or training.

Outside of peer review, research has been performed recently to mine wishes and suggestions in product reviews and political discussion. Goldberg *et al.* (2009) analyzed the WISH[3] corpus and built wish detectors based on simple word cues and templates. Focusing on product reviews only, Ramanand *et al.* (2010) created two corpora of suggestion wishes (wishes for a change in an existing product or service) and purchasing wishes (explicit expressions of a desire to purchase a product), and developed rules for identifying wish sentences from non-wish ones. Both (Goldberg et al., 2009; Ramanand et al., 2010) created rules manually by examining the data. Although we hypothesize that wishes are related to solutions in peer review, our educational data makes direct application of product-motivated rules difficult. We thus currently use statistical machine learning for our initial research, but plan to explore incorporating expression rules to enhance our model.

Sub-sentence annotation has gained much interest in sentiment analysis and opinion mining. One notable work is (Wilson et al., 2005) in which the author addressed the problem that the contextual polarity (i.e. *positive*, *negative*, or *neutral*) of the phrase in which a word appears may be different from the word's prior polarity. We will also use a phrase-level annotation, as described below.

3 Argument diagram review corpus

Diagramming software tools such as LASAD (Scheuer et al., 2010) are increasingly being used to teach student argumentation skills through graphical representations. Graphical argument environments typically allow students to create diagrams in which boxes represent statements and links represent argumentative or rhetorical relations. This helps students focus on abstract argu-

[2]Cf. (Cho, 2008) "a self-contained message on a single piece of strength or weakness found in peer writing."

[3]http://www.timessquarenyc.org/nye/nye_interactive.html

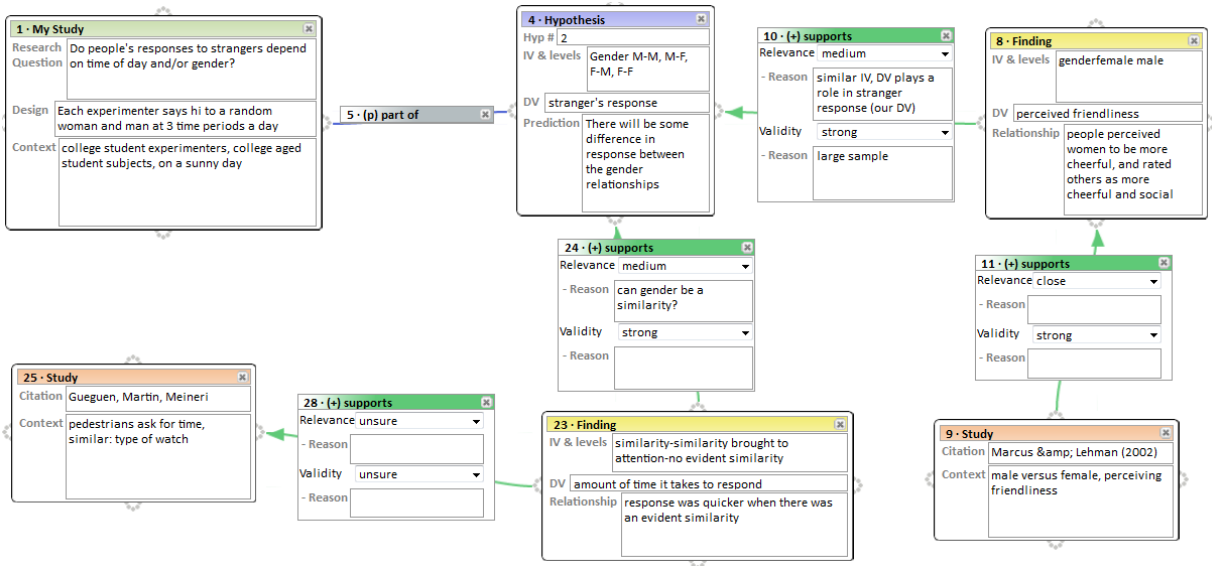

Figure 2: Part of a student argument diagram.

ment schemes before learning how to write argumentative essays. To further help students create good argument diagrams, it has recently been suggested that receiving and providing feedback on argument diagrams might yield useful pedagogical benefits (Falakmassir et al., July 2013), analogously to improving writing via peer review of papers.

Our corpus consists of a subset of comments from diagram reviews collected from nine separate sections of an undergraduate psychology course. Student argument diagrams were created using an instructor-defined diagram ontology. The diagram ontology defines five different types of nodes: *Current study, Hypothesis, Theory, Finding*, and *Study (for reference)*. The ontology also defines four different types of arcs that connect nodes: *Supports, Opposes, Part-of*, and *Undecided*. Figure 2 shows part of a student argument diagram that includes two studies, each of which supports a finding which in turn supports or opposes a hypothesis. In the course that generated our corpus, students first created graphical argument diagrams using LASAD to justify given hypotheses. Student argument diagrams were then distributed, using the SWoRD (Cho and Schunn, 2007) web-based peer-review system, to other students in the class for reviewing. Student authors potentially revised their argument diagrams based on peer feedback, then used the diagrams to write the introduction of associated papers. Diagram reviews consist of multiple written feedback comments in response

<**IU**> <Pr>*Not all of these field are filled out, which makes it hard to get a clear idea of how legit these studies are.*</Pr> <**/IU**> <**IU**> <Pr>*Also, some are unclear. An example is 24-supports where the reason is a question.*</Pr> <Sl>**I think there should be a substantial reason there instead of a question to convince me why it is relevant.**</Sl> <**/IU**>

Table 1: Example of an annotated comment. Markers <IU>: idea unit, <Sl>: solution, <Pr>: problem. Problem text is italic and solution text is bold for illustration purpose.

to rubric prompts, i.e. review dimensions. Student reviewers were required to provide at least one but no more than three comments for each of five review dimensions. Figure 1 shows three sample peer comments for three review dimensions (i.e. dimensions 1, 3 and 5).

Following prior work on peer review analysis (Lippman et al., 2012; Nguyen and Litman, July 2013), the first author composed a coding manual for peer reviews of argument diagrams. An annotator first segments each comment into idea units (defined as contiguous feedback referring to a single topic). Note that idea-unit segmentation is necessary to make coding reliable. We however do not exploit idea unit information for our current prediction tasks. Then the annotator codes each idea unit for different features among which *solution* and *problem* are

Label	Number of comments
`Solution`	178
`Problem`	194
`Combined`	135
`Non-criticism`	524
Total	1031

Table 2: Comment label distribution.

the two labels used in this study. These labels are then used to assign a **feedback type** (i.e. `solution`, `problem`, `combined`, and `non-criticism`) to the comment as a whole. The comment is labeled `Solution` if at least one of its idea units presents a solution but no problem unit is explicitly present. If no solution idea is found, the comment is labeled `Problem` if at least one of its idea units presents a problem. The comment is labeled `Combined` if it has both solution and problem idea units, or `Non-criticism` if it does not have solution or problem. `Non-criticism` units can be praise, summary or text that does not express any idea, e.g. *"Yes, it is."* Table 1 shows an example annotated feedback comment that consists of two idea units. The first idea unit is about *empty fields*, and the second is about *reason is a question*. Based on the annotations shown, the comment as a whole has the label `Combined`.

We had one undergraduate psychology major annotate the 1031 comments in our corpus, yielding the label distribution shown in Table 2. The first author also annotated 244 randomly selected comments, solely to evaluate inter-coder agreement. The obtained agreement of comment labels was high, with accuracy 0.81 and kappa 0.74.

In addition to comment labeling, the annotator also highlighted[4] text spans that explain the labels. The marked text span must either express solution or problem information but cannot express both. Therefore we require the annotator to highlight at the phrase (i.e. sub-sentence) level, and that each marked text must be completely within an idea unit. Generally speaking this requirement does not increase cognitive workload because annotators already have to read the comment and notice any solution or problem mentioned before labeling.

[4]Highlighting was made possible using macros in Microsoft Word. Annotators select the text of interest, then click a button corresponding to the relevant label, e.g. problem.

Label	Sentence
Problem	Not all of these field are filled out, which makes it hard to get a clear idea of how legit these studies are.
Problem	Also, some are unclear.
Problem	An example is 24-supports where the reason is a question.
Solution	I think there should be a substantial reason there instead of a question to convince me why it is relevant.

Table 3: Examples of labeled sentences extracted from the annotated comment.

Category	Number of sentences
Solution	389
Problem	458
Non-criticism	1061
Total	1908

Table 4: Sentence label distribution.

Although we asked the annotator to mark text spans which convey problem or solution information, we did not ask the annotator to break each text span into sentences. The first reason is that the problem or solution text might only be part of a sentence and highlighting only the informative part will give us more valuable data. Second, sentence segmentation can be performed automatically with high accuracy. After the corpus was annotated, we ran a sentence segmentation procedure using NLTK[5] to create a labeled corpus at the sentence level as follows. Each comment is broken into three possible parts: *solution* including all solution text marked in the comment, *problem* including all problem text, and *other* for non-criticism text. Each part is then segmented into sentences and each sentence is assigned the label of the part to which it belongs. It may happen that the segmented text is a phrase rather than a complete sentence. We consider such phrases as reduced sentential-like text, and we use the term *sentence(s)* to cover such sub-sentence forms, as well. Labeled sentences of the comment in Table 1 are shown in Table 3. After discarding empty sentences and those of length 1 (all of those are in the *non-criticism* category), there are 1908 sentences remaining, distributed as shown in Table 4.

[5]www.nltk.org

4 Experimental setup

Sections 6 and 7 report the results of two different experiments involving the prediction of feedback types at the comment level. While each experiment differs in the exact classes to be predicted, both compare the predictive utility of the same two different model-building approaches:

- *Trained using comments* (CTRAIN): our baseline[6] approach learns comment prediction models using labeled feedback comments for training.
- *Trained using sentences* (STRAIN): our proposed approach learns sentence prediction models using labeled sentences, then aggregates sentence prediction outputs to create comment labels. For example, the aggregation used for the experiment in Section 6 is as follows: if at least one sentence is predicted as *Solution/Problem* then the comment is assigned *Solution/Problem*.

We hypothesize that the proposed approach will yield better predictive performance than the baseline because the former takes advantage of cleaner and more discriminative training data.

To make the features of the two approaches comparable, we use the same set of generic linguistic features:

- Ngrams to capture word cues: word unigrams, POS/word bigrams, POS/word trigrams, word and POS pairs, punctuation, word count.
- Dependency parse to capture structure cues.

We skip domain and course-specific features (e.g. review dimensions, diagram keywords like *hypothesis*) in order to make the learned model more applicable to different diagram review data. Instead, we search for diagram keywords in comments and replace them with the string "KEYWORD". The keyword list can be extracted automatically from LASAD's diagram ontology. Adding metadata features such as comment and sentence ordering did not seem to improve performance so we do not include such features in the experiments below.

[6]The use of comment-level annotations for training and testing is similar to (Nguyen and Litman, July 2013).

Following (Xiong et al., 2012), we learn prediction models using logistic regression. However, in our work both feature extraction and model learning are performed using the LightSide[7] toolkit. As our data is collected from nine separate sections of the same course, to better evaluate the models, we perform cross-section evaluation in which for each fold we train the model using data from 8 sections and test on the remaining section. Reported results are averaged over 9-fold cross validations. Four metrics are used to evaluate prediction performance. Accuracy (Acc.) and Kappa (κ) are used as standard performance measurements. Since our annotated corpus has imbalanced data which makes the learned models bias to the majority classes, we also report the Precision (Prec.) and Recall (Recl.) of predicting the minor classes.

5 Sentence prediction performance

We first evaluate models for predicting binary versions of the sentence labels from Table 4 (e.g. solution or not), as this output will be aggregated in our proposed STRAIN approach. The results of using sentence training (STr) and sentence testing (STe) are shown in the STR/STE row of Table 5. For comparison, the first row of the table shows the performance of a majority baseline approach (MAJOR), which assigns all sentences the label of the relevant major class in each prediction task. To confirm that a sentence-level annotated corpus is necessary to train sentence prediction models, a third approach that uses labeled comment data for training (CTr) but sentences for testing (STe) is included in the CTR/STE row. As we can see, STR/STE models outperform those of CTR/STE and MAJOR for all 4 metrics[8]. The comment versus sentence training yields significant differences for predicting *Problem* and *Criticism* sentences.

6 Three feedback type prediction tasks

In this experiment we evaluate our hypothesis that STRAIN outperforms CTRAIN by comparing performance on three feedback type prediction tasks at the comment level (derived from Table 2):

- *Problem v. Non-problem*. The *Problem* class includes problem and combined comments.

[7]http://ankara.lti.cs.cmu.edu/side/download.html

[8]Note that κ in general, and precision and recall of minor classes, are not applicable when evaluating MAJOR.

	Solution				*Problem*				*Criticism*			
Model	Acc.	κ	Prec.	Recl.	Acc.	κ	Prec.	Recl.	Acc.	κ	Prec.	Recl.
MAJOR	0.80	-	-	-	0.76	-	-	-	0.56	-	-	-
CTR/STE	0.87	0.57	0.70	0.62	0.75	0.22	0.48	0.29	0.75	0.48	0.76	0.63
STR/STE	0.88	0.61	0.76	0.63	**0.81**	**0.44**	**0.62**	**0.51**	*0.80*	*0.59*	*0.79*	*0.74*

Table 5: Prediction performance of three tasks at the sentence level. Comparing STR/STE to CTR/STE: *Italic* means higher with $p < 0.05$, **Bold** means higher with $p < 0.01$.

- *Solution v. Non-solution*. The *Solution* class includes solution and combined comments.
- *Criticism v. Non-criticism*. The *Criticism* class includes problem, solution and combined comments.

The two approaches are also compared to majority baselines (MAJOR) and a hybrid approach (HYBRD) that trains models using labeled sentence data but tests on labeled comments.

As shown in Table 6, both MAJOR and HYBRD perform much worse than CTRAIN and STRAIN. We note that while HYBRD gives comparably high precision, its kappa and recall do not match those of CTRAIN and STRAIN. Comparing CTRAIN and STRAIN, the results confirm our hypothesis that STRAIN outperforms CTRAIN. The major advantage of STRAIN is that it only needs one correctly predicted sentence to yield the correct comment label. This is particularly beneficial for predicting problem comments, where the improvement is significant for 3 of 4 metrics.

As our evaluation is cross-section, folds do not have identical label distributions. Therefore we look at prediction performance for each of the nine individual sections. We find that the sentence level approach yields higher performance on all four metrics in six sections when predicting both *Solution* and *Problem* task, but only two sections for *Criticism*. For the *Criticism* task – where it is not necessary to exclusively differentiate between *Solution* and *Problem*, training prediction models using labeled sentences does not yield higher performance than the traditional approach.

Roughly comparing predicting at the sentence level (Table 5) versus the comment level (Table 6), we note that the sentence level tasks are more difficult (e.g. lower absolute kappas) despite an intuition that the labeled sentence corpus is cleaner and more discriminative compared to the labeled comment corpus. The observed performance disparity shows the necessity of developing better sentence prediction models, which we leave to future work.

7 A case study experiment

To the best of our knowledge, (Xiong et al., June 2010; Xiong et al., 2012) contain the only published models developed for predicting feedback types. A comment-level solution prediction model has since been deployed in their peer review software to evaluate student reviewer comments in classroom settings, using the following 3-way classification algorithm[9]. Each student comment is classified as either a criticism (i.e. presents problem/solution information) or a non-criticism. The non-criticism comment is labeled *NULL*. The criticism comment is labeled *SOLUTION* if it contains solution information, and labeled *PROBLEM* otherwise.

To evaluate our proposed STRAIN approach in their practically-motivated setting, we follow the description above to relabel peer feedback comments in our corpus to new labels: *NULL, PROBLEM*, and *SOLUTION*. We also asked the authors of (Xiong et al., 2012) for access to their current model and we were able to run their model on our feedback comment data. While it is not appropriate to directly compare model performance as Xiong *et al.* were working with paper (not diagram) review data, we report their model output, named PAPER, to provide a reference baseline. We expect the PAPER model to work on our diagram review data to some extent, particularly due to its predefined seed words for solution and problem cues. Our CTRAIN baseline, in contrast, trains models regarding the new label set using relabeled diagram comment data, with the same features and learning algorithm from the prior sections. The majority baseline, MAJOR, assigns all comments the major class label (which is now *NULL*).

Regarding our STRAIN sentence level ap-

[9]Personal communication.

	Solution				*Problem*				*Criticism*			
Model	Acc.	κ	Prec.	Recl.	Acc.	κ	Prec.	Recl.	Acc.	κ	Prec.	Recl.
MAJOR	0.70	-	-	-	0.68	-	-	-	0.51	-	-	-
HYBRD	0.82	0.52	0.87	0.48	0.75	0.36	0.68	0.41	0.78	0.56	0.84	0.68
CTRAIN	0.87	0.67	0.84	0.71	0.76	0.43	0.65	0.55	0.83	0.66	0.85	0.80
STRAIN	0.88	*0.71*	0.86	*0.74*	**0.81**	**0.55**	*0.71*	**0.66**	0.85	0.70	0.84	*0.85*

Table 6: Prediction performance of three tasks at comment level. Comparing STRAIN to CTRAIN: *Italic* means higher with $p < 0.1$, **Bold** means higher with $p < 0.05$.

1. For each sentence, label it *SOLUTION* if it is predicted as *Solution* by the *Solution* model.
2. For a predicted *Non-solution* sentence, label it *NULL* if it is predicted as *Non-criticism* by the *Criticism* model.
3. For a predicted *Criticism* sentence, label it *PROBLEM* if it is predicted as *Problem* by the *Problem* model.
4. For a predicted *Non-problem* sentence, label it *SOLUTION*

Table 7: Relabel procedure.

Model	Acc.	κ
MAJOR	0.51	-
PAPER	0.71	0.49
CTRAIN	0.76	0.60
RELABELFIRST	0.79	0.66
INFERFIRST	0.79	0.66

Table 8: Prediction performance of different approaches in a case study.

proach, we propose two aggregation procedures to infer comment labels given sentence prediction output. In the first procedure, RELABELFIRST, we infer new sentence labels regarding *NULL, PROBLEM*, and *SOLUTION* using a series of conditional statements. The order of statements is chosen heuristically given the performance of individual models (see Table 5) and is described in Table 7. Given the sentences' inferred labels, the comment is labeled *SOLUTION* if it has at least one *SOLUTION* sentence. Else, it is labeled *PROBLEM* if at least one of its sentences is *PROBLEM*, and labeled *NULL* otherwise. Our second aggregation procedure, called INFERFIRST, follows an opposite direction in which we infer comment labels regarding *Solution, Problem*, and *Criticism* before re-labeling the comment regarding *SOLUTION, PROBLEM*, and *NULL* following the order of conditional statements in the relabel procedure.

As shown in Table 8, the MAJOR and PAPER models perform much worse than the other three models. While the PAPER model has accuracy close to that of the other models, its kappa is far lower. Regarding the three models trained on diagram review data, the two sentence level models outperform the CTRAIN model. Particularly, kappas of the two sentence level models are either significantly higher (for INFERFIRST) or marginally higher (for RELABELFIRST) compared to kappa of CTRAIN. To further investigate performance disparity between models, we report in Table 9 precision and recall of different models for each class. The PAPER model achieves high precision but low recall for *SOLUTION* and *PROBLEM* classes. We reason that the model's seed words help its precision, but its ngram features, which were trained using paper review data, cannot adequately cover positive instances in our corpus. The two sentence level models perform better for the *PROBLEM* class than the other two models, which is consistent with what is reported in Table 6. Comparing the two sentence level models, INFERFIRST better balances precision and recall than RELABELFIRST.

8 The sentence level is right

The experimental results in the previous two sections have demonstrated that sentence prediction output helps improve prediction performance at the comment level. This supports our hypothesis that sentence prediction is the right level for enhancing peer review systems to detect and respond to multi-sentence review comments of low quality. In our labeled sentence corpus, each instance either expresses a solution, a problem, or is a non-criticism, so the data is cleaner and more discrim-

	SOLUTION		*PROBLEM*		*NULL*	
Model	Prec.	Recl.	Prec.	Recl.	Prec.	Recl.
MAJOR	-	-	-	-	0.51	1.00
PAPER	0.81	0.62	0.58	0.29	0.70	0.92
CTRAIN	0.84	0.75	0.55	0.41	0.78	0.90
RELABELFIRST	0.72	0.90	0.66	0.48	0.88	0.84
INFERFIRST	0.75	0.86	0.61	0.55	0.88	0.84

Table 9: Precision and recall of different models in a case study.

inative than the labeled comment corpus. This is a nice property that helps reduce feature collocation across exclusive classes, *Problem vs. Solution* for example, which is a danger of training on feedback comments due to *Combined* instances. Moreover, our annotated comment corpus has solution and problem text marked at the sub-sentence level, which is a valuable resource for learning solution and problem patterns and linguistic cues.

Improving peer feedback prediction accuracy is not the only reason we advocate for the sentence level. We envision that the sentence level is the necessary lower bound that a peer review system needs to handle new advanced functionalities such as envisioned in Figure 1. Being able to highlight featured text in a peer comment is a useful visualization function that should help peer reviewers learn from live examples, and may also help student authors quickly notice the important point of the comment.

Sentence and phrase level annotation is made easy with the availability of many text annotation toolkits; BRAT[10] (Stenetorp et al., 2012) is an example. From our work, marking text spans by selecting and clicking requires a minimal additional effort from annotators and does not cause more cognitive workload. Moreover, we hypothesize that through highlighting the text, an annotator has to reason about why she would choose a label, which in turn makes the annotation process more reliable. We plan to test whether annotation performance does indeed improve in future work.

9 Conclusions and future work

In this paper we present a sentence-level annotated corpus of argument diagram peer review data, which we use to develop comment-level predictions of peer feedback types. Our work is the first of its kind in building an automated feedback type assessment component for reviews of argument diagrams rather than papers. We have demonstrated that using sentence prediction outputs to label the corresponding comments outperforms the traditional approach that learns models using labeled comments. The improvement of using sentence prediction outputs is more significant for more difficult tasks, i.e. *Problem vs. Non-problem*, in which textual expression varies greatly from explicit to implicit. In a case study mimicking a real application setting to experiment with the proposed models, we achieved a similar verification of the utility of sentence models. Given our imbalanced training data labels and our avoidance of using domain-specific features, these first results of our two experiments are promising.

In these first studies, our models were trained using generic prediction procedures, e.g., using basic linguistic features without feature selection or tuning. Thus our next step is to analyze prediction features for their predictiveness. We also plan to incorporate human-engineered rules for solution and problem text. We aim to improve performance while keeping feature generality. An interesting experiment we may conduct is to test our learned models on paper review data to evaluate performance and generality in an extreme setting.

Acknowledgments

This work is supported by NFS Grant No. 1122504. We are grateful to our colleagues for sharing the data. We thank Kevin Ashley, Wencan Luo, Fan Zhang, other members of the Argument-Peer and ITSPOKE groups as well as the anonymous reviewers for their valuable feedback.

References

Kwangsu Cho and Christian D. Schunn. 2007. Scaffolded writing and rewriting in the discipline: A web-based reciprocal peer review system. *Computers and Education*, 48(3):409–426.

[10] http://brat.nlplab.org/

Kwangsu Cho. 2008. Machine classification of peer comments in physics. In *Proceedings 1st international conference on Educational Data Mining (EDM)*, pages 192–196.

Mohammad Falakmassir, Kevin Ashley, and Christian Schunn. July 2013. Using argument diagramming to improve peer grading of writing assignments. In *Proceedings of the 1st Workshop on Massive Open Online Courses at 16th International Conference on Artificial Intelligence in Education (AIED), Memphis, TN*, pages 41–48.

Sarah Gielen, Elien Peeters, Filip Dochy, Patrick Onghena, and Katrien Struyven. 2010. Improving the effectiveness of peer feedback for learning. *Learning and Instruction*, 20(4):304–315.

Andrew B. Goldberg, Nathanael Fillmore, David Andrzejewski, Zhiting Xu, Bryan Gibson, and Xiaojin Zhu. 2009. May all your wishes come true: A study of wishes and how to recognize them. In *Proceedings Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, NAACL-HTL*, pages 263–271.

Jordan Lippman, Mike Elfenbein, Matthew Diabes, Cori Luchau, Collin Lynch, Kevin Ashley, and Chris Schunn. 2012. To revise or not to revise: What influences undergrad authors to implement peer critiques of their argument diagrams? In *International Society for the Psychology of Science and Technology 2012 Conference*. Poster.

Melissa M. Nelson and Christian D. Schunn. 2009. The nature of feedback: how different types of peer feedback affect writing performance. *Instructional Science*, 37(4):375–401.

Huy V. Nguyen and Diane J. Litman. July 2013. Identifying localization in peer reviews of argument diagrams. In *Proceedings 16th International Conference on Artificial Intelligence in Education (AIED), Memphis, TN*, pages 91–100.

Huy Nguyen, Wenting Xiong, and Diane Litman. June 2014. Classroom evaluation of a scaffolding intervention for improving peer review localization. In *Proceedings 12th International Conference on Intelligent Tutoring Systems (ITS), Honolulu, HI*, pages 272–282.

Lakshmi Ramachandran and Edward F. Gehringer. 2011. Automated assessment of review quality using latent semantic analysis. In *Proceedings 11th IEEE International Conference on Advanced Learning Technologies (ICALT)*, pages 136–138.

J. Ramanand, Krishna Bhavsar, and Niranjan Pedanekar. 2010. Wishful thinking: finding suggestions and 'buy' wishes from product reviews. In *Proceedings the NAACL-HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, pages 54–61.

Oliver Scheuer, Frank Loll, Niels Pinkwart, and Bruce M. McLaren. 2010. Computer-supported argumentation: A review of the state of the art. *International Journal of Computer-Supported Collaborative Learning*, 5(1):43–102.

Pontus Stenetorp, Sampo Pyysalo, Goran Topic, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. Brat: a web-based tool for nlp-assisted text annotation. In *Proceedings the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Joint Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing (HLT-EMNLP)*, pages 347–354.

Wenting Xiong and Diane Litman. 2010. Identifying problem localization in peer-review feedback. In *Proceedings 10th International Conference on Intelligent Tutoring System (ITS), Pittsburgh, PA*. Poster.

Wenting Xiong, Diane Litman, and Christian Schunn. 2012. Natural language processing techniques for researching and improving peer feedback. *Journal of Writing Research*, 4(2):155–176.

Wenting Xiong, Diane Litman, and Christian Schunn. June 2010. Assessing reviewer's performance based on mining problem localization in peer-review data. In *Proceedings 3rd International Conference on Educational Data Mining (EDM), Pittsburgh, PA*, pages 211–220.