Automated Software Engineering manuscript No.
(will be inserted by the editor)

Grouping Related Stack Overflow Comments for Software Developer Recommendation

Viral Sheth · Kostadin Damevski

Received: date / Accepted: date

Abstract Stack Overflow is a question and answer forum widely used by developers all over the world. Contributors share their knowledge on this platform not only in the form of answers, but also as comments to those answers. With millions of developer-contributed comments, the valuable knowledge contained within them remains difficult to locate by readers. Moreover, Stack Overflow's comment hiding mechanism that only shows the top five most highly voted comments and hides the remaining leads to wealth condensation. Recently, researchers have observed that the Stack Overflow's comment display mechanism hides important and relevant comments and makes it difficult for readers to understand the conversational context, as many comments are related to other hidden comments.

In this paper, we propose a set of features and a machine learning-based technique to identify the relatedness of pairs of comments. Further, we extend the relatedness into comment clustering, as, with clusters, readers can get the entire context of a set of comments that form a single conversational thread. We evaluate our methods against several baselines to show that they provide strong improvements, although the problem in general is made difficult by the short text and narrow topic of discussion in the comments.

Keywords software developer discussions, developer forums, Stack Overflow, comment grouping, comment ranking

V. Sheth
Department of Computer Science
Virginia Commonwealth University
Richmond, VA 23284, U.S.A.
E-mail: shethvh@vcu.edu

K. Damevski
Department of Computer Science
Virginia Commonwealth University
Richmond, VA 23284, U.S.A.
E-mail: kdamevski@vcu.edu

1 Introduction

Stack Overflow [4], the most popular website of the Stack Exchange Network [2], consists of free question and answer forums on a wide set of software development-related tools and technologies. Stack Overflow provides a platform where users can post their questions pertaining to programming-related problems they face, while other users provide answers. At the time of writing, Stack Overflow has grown to over 20 million questions and 31 million answers contributed by a user-base of 13 million developers worldwide [5]. Contributing Stack Overflow users provide novel answers to newly posed or existing questions on the platform. Other users in the community who find a specific answer useful can award an upvote, which provides 10 reputation points to the user who contributed the answer. Conversely, a poorly written answer attracts downvotes, which reduces the total reputation points of the contributing user by 2 points. After a period of time, the total number of upvotes minus the number of downvotes on a given answer are extremely good indicators of the quality of an answer [21, 28].

As shown in Figure 1, in addition to providing a question or answer Stack Overflow users may also choose to post comments to a specific question or answer. The purpose of each comment is to seek or provide clarification to an ambiguous question or answer, to provide supplemental details to an incomplete answer, or to simply praise a well-written question or answer [13]. Insightful comments may also attract votes from Stack Overflow users, reflecting their quality and usefulness to the community. Researchers have recently observed that Stack Overflow comments often contain valuable knowledge that

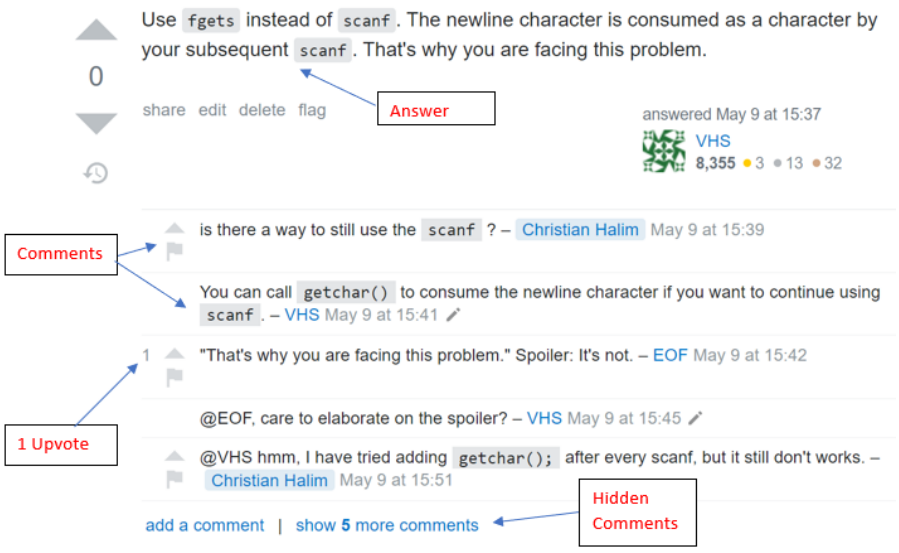


Fig. 1 Example of a Stack Overflow answer and associated visible and hidden comments.

can aid software developers [24]. However, comments are often overlooked by many Stack Overflow users as sources of solutions to their problems [31].

At present, there are more than 78 million comments on Stack Overflow [3]. Due to the sheer volume of the comments, readers usually tend to neglect reading the comments as they are focused primarily on reading the questions and answers [26]. To highlight comments relevant to readers, Stack Overflow has a comment hiding mechanism that shows only the top 5 most highly voted comments on each question or answer and hides the rest. If multiple comments have the same number of votes, comments with older timestamps are shown and when a question reaches a threshold of 30 answers, all comments with zero votes on both the question and all its answers are collapsed and hidden [1].

One of the problems with the top N system for comment visibility is wealth condensation [10]. That is, newer comments lack votes and are not displayed among the top 5, which leads to them not getting as many votes, which in turn makes it even harder to getting displayed among the top 5. Readers have to click the *"show ⟨x⟩ more comments"* link to display the hidden comments. Prior research has shown that there are a lot of examples where the hidden comments could add value to a reader, by providing additional information, clarifying a point, or simply asking a question so no one else needs to [12]. Second, comments often form a conversational thread. In order to understand the context of the discussion between multiple groups of developers on a given post, one has to painstakingly scroll through and read all the comments. It is often observed in conversations involving Q&A that the answerer's comments get more upvotes than the questioner's comments [27]. Therefore, the comment hiding mechanism which hides the low votes comments makes it difficult for the readers to understand the conversational context. Because of the abundance of comments as well as the comment hiding mechanism, useful discussion and knowledge are often not easily available to the users [30].

In this paper, we first describe a technique to identify related pairs of comments and measure the strength of their relationship. By measuring comment relatedness, we can provide *a ranking of additional hidden comments a reader could be interested in if she is reading a specific visible comment.* Second, we extend the comment relatedness measure to *identify related groups (or clusters) of comments*, which focus on a specific salient topic. By grouping comments in this way we enable readers to expand from a single visible comment to understand conversational context or more deeply explore a specific topic. Alternatively, if the comment is on a topic that they don't want to read, move away from the group to explore other topics. To summarize, the contributions of this paper are:

- an approach, based on a novel set of features, to assess the strength of the relationship between selected pairs of Stack Overflow comments posted on a single question or answer;
- clustering technique that forms groups from the given set of comments on a Stack Overflow post;

– evaluation of the effectiveness of the comment relatedness and clustering using a manually annotated corpus.

The remainder of this paper is organized in the following way. Section 2 describes a motivational study of relatedness in Stack Overflow comments. In Section 3, we focus on automatically determining comment relatedness. We provide details of our approach based on machine learning to compute relatedness score of a pair of comments as well as the graph cutting algorithm that we used to form groups from the given set of comments. Section 4 illustrates how we we have evaluated our methods, while Section 5 shows the results of our evaluation. Section 7 highlights prior research work related to the Stack Overflow comments. Finally, in Section 8, we conclude the paper and discuss avenues for future work.

2 Motivation: Comment Relatedness Study

We started our research with the goal of understanding if certain Stack Overflow comments are highly related and identifying the salient characteristics that can be used to determine if a pair of comments is related. Using this information, a Stack Overflow user that is reading one of the comments can be recommended another highly related comment (or a highly unrelated one if the reader wants to change topics). Alternatively, by understanding how groups of comments are related, Stack Overflow's interface may be modified to display or hide groups of comments, rather than displaying a selection of comments solely on their vote totals. To this end, we began by conducting a human study focused on software developers' perception of Stack Overflow comment relatedness.

Procedure. Using our personal networks we invited via e-mail several professionals from industry as well as graduate and undergraduate students to participate in our study. Twelve individuals with diverse experience and background agreed to participate (6 graduate students and 6 from industry). We informed all participants that the study was about Stack Overflow comments, but not the specifics of what we intended to study about the comments. All the participants indicated that they were familiar with Stack Overflow and regularly visited the site to learn information. We followed IRB guidance in informing the participants that their participation was voluntary and could be terminated at any point, that the survey would collect only their basic demographics, and that we would protect the participants' confidentiality.

Data. We created an online survey consisting of a set of questions asking the study participants to select a relatedness level between a pair of Stack Overflow comments, with three relatedness levels to choose from: Unrelated, Slightly Related and Strongly Related. We explained the definition of these levels to the participants as, Unrelated corresponds to no observable relationship between the comments, Strongly Related corresponds to one of the comments needing the other for comprehension, and Slightly Related as two comments that have a relationship but can be easily understood in isolation. Each participant

answered the question for 40 pairs of Stack Overflow comments. In total, we collected ratings for 240 pairs of comments across six surveys.

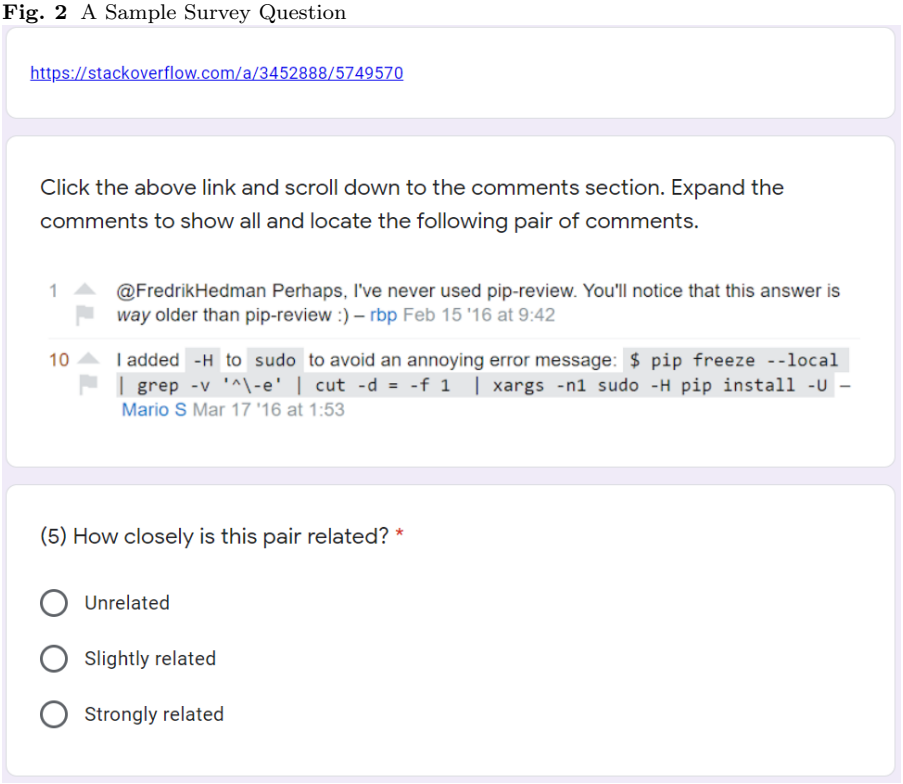
We were specifically interested in the relationship between pairs of comments where one appears in the top 5 comments (i.e., the original, un-expanded set) on a Stack Overflow post and the other comment does not. In order to ensure that these pairs of comments are selected systematically, in a way that maximizes the selection of most likely to be replaceable pairs for evaluation by the study participants, we devised the following Stack Overflow comment selection process:

1. *Question Selection:* We focused on Stack Overflow questions that are tagged with the `python` tag, as we assumed most of our participants will be familiar with this programming language. We selected the most recent set of Stack Overflow questions tagged with this tag, sorted by their vote count to ensure high quality.
2. *Post Selection:* Given a Stack Overflow question from the previous step, we selected up to 3 posts (i.e., question or answers) that had the highest votes and at least 7 comments. The limit of 3 posts was to ensure we did not oversample from a single question and the vote counts were used to ensure high quality posts. At the time of writing, 2,402,771 posts or 67% of posts with hidden comments (around 9% of all Stack Overflow posts) contain 7 or more comments, representing a significant proportion of the sample space. The reason we selected a threshold of 7 or more comments has to do with having sufficient number of comments for the comment selection process, described next.
3. *Comment Selection:* For each Stack Overflow post identified in the previous step, we selected the comment that was the "last one in" from the top 5 comments that are shown with each post as our *seed comment*. We focused on this comment as it was the one most interesting to study since it could have easily not made it in the top 5. If another comment is added with just 1 vote more than this one, it wouldn't get displayed in the top 5 anymore. Similarly, if there was just one more older comment with the same number of votes as this one, this comment wouldn't have made it to the visible comments list. The other interesting aspect of the 5th comment is that it often does not possess an excessive number of votes; the comment votes usually form an exponential distribution, i.e., the newer comments tend to attract less votes. Since the older comments get displayed over the newer ones, they are more likely to get votes. In addition, the 5th comment usually is neither the oldest nor the newest comment appearing on the post. In other words, this seed comment usually has a preceding and a succeeding comment for comparison, which ensures that our comment pair selection process, described next, is consistent across all posts. Using seed utterances and previous and successive instances is a strategy used by other researchers analyzing similar, conversational data [25].
4. *Comment Pair Selection:* Using the seed comment selected in the previous step, we created two pairs of comments for the study. One pair consists of

the seed comment and the first comment that occurred prior to it in time that did not appear in the top 5 comments. The second pair was the seed comment and the one immediately following it, in time, that was not part of the top 5 comments, i.e., it was hidden by default.

The 6 surveys were divided among the 12 participants in such a way that each survey is taken by exactly two participants. The surveys explained them the task at hand with examples in detail. After taking the participants' agreement that they have understood the task, the survey presented them with the forty questions, one question at a time. A sample survey question asking the user to annotate the relatedness between a pair of Stack Overflow comments is shown in the figure 2.

Fig. 2 A Sample Survey Question



Study Results. The annotation data was collected for each of the six surveys for each participant and analyzed for patterns. We note that only 25 out of the 240 pairs (10.42%) were annotated by the two participants with Unrelated and Strongly Related. This indicates that the participants did not often disagree in the strongest way. Complete agreement between the participants on the annotations was high; for 147 out of a total of 240 pairs (61.25%) of comments

both participants annotated the pairs with the same relatedness level. We also computed Cohen's Kappa to measure the agreement between the participants with an average Cohen's Kappa value of 0.40 for the six surveys. According to [16], a value between 0.21 and 0.40 indicates a fair agreement. Thus, the survey results clearly revealed that there was a reasonable level of agreement among the participants for judging the relatedness levels between pairs of comments.

There was a significant number of comments that our annotators found to be related. More specifically, 102/240 (42.50%) had at least one participant and 48/240 (20.00%) had both participants annotating them with 'Strongly Related' label. Figure 3 shows an example pair of comments annotated as 'Strongly Related' by both participants. We observe that the second comment, while voted 97 times (and appearing in the top 5), is a response to the former comment which has no votes (and does not appear in the top 5 comments).

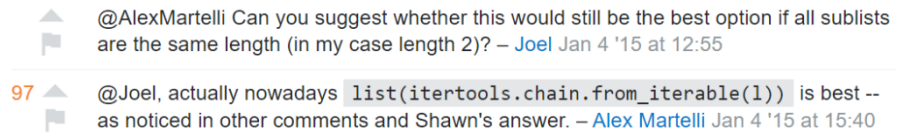


Fig. 3 An example pair of comments annotated as 'Strongly Related' by both participants

There were 93/240 (38.75%) pairs which were annotated with at least one 'Slightly Related' label and 28/240 (11.66%) with both participants agreeing on this label. Figure 4 shows an example pair of comments annotated as 'Slightly Related' by both participants.

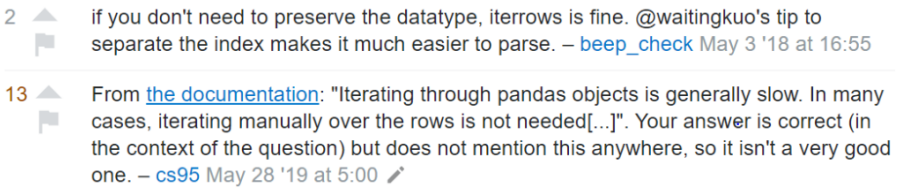


Fig. 4 An example pair of comments annotated as 'Slightly Related' by both participants

Finally, the largest proportion of comments pairs, i.e., 132/240 (55.00%) had at least one and 71/240 (21.58%) had both participants annotating them with the 'Unrelated' label. Figure 5 shows an example pair of comments annotated as 'Unrelated' by both participants.

Next, we analyzed the survey data to identify certain characteristics that can potentially influence the relationship of the Stack Overflow comments with respect to relatedness, especially with respect to the 'Strongly Related' label.

– Out of the total of 240 pairs of comments, 52 pairs had author of one comment refer to the author of another (using the @author mechanism).

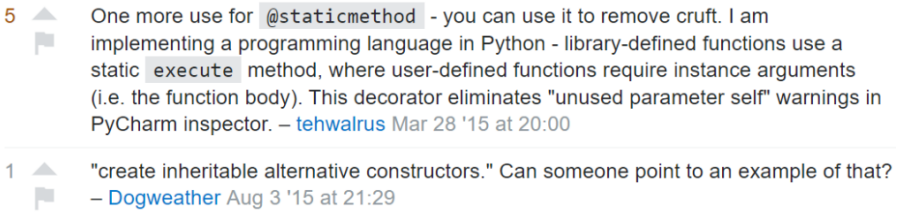


Fig. 5 An example pair of comments annotated as 'Unrelated' by both participants

The survey data indicated that 45/52 (86.54%) pairs had at least one participant annotating them with 'Strongly Related' label.
- There were 7 pairs of comments that had the same author. The survey data showed that 5/7 pairs (71.43%) were annotated with at least one 'Strongly Related' label.
- We observed 3 pairs of comments were such that their respective authors were referring to a third author using Stack Overflow's @author mechanism. All 3/3 (100%) were annotated with at least one 'Strongly Related' label.
- The dataset contained 22 pairs of comments that were posted on the same day, while 19/22 (86.36%) of which were annotated with at least one 'Strongly Related' label. Similarly, 15 pairs of comment were posted within a week but not the same day. Out of these 12/15 (80%) were annotated with at least one 'Strongly Related' label. Finally, 203 pairs of comments were more than a week apart. However, out of these, 71/203 (34.98%) pairs were annotated with at least one 'Strongly Related' label.

This clearly indicates that authorship and time are important factors in determining the relatedness of comments. A pair of comments is likely to be strongly related if it is posted within a shorter time frame and/or if it contains explicit references to the authors. Table 2 summarizes how the aforementioned characteristics between pairs of comments predict their strong relationship.

Table 1 Proportions of comments annotated as 'Strongly Related' exhibiting different characteristics.

Characteristic	**Comments rated as 'Strongly Related'** ***(by at least one participant)***
Same Author	5/7 (71.43%)
Refers to the Author of the Other Comment	45/52 (86.54%)
Both Refer to a Common Third Author	3/3 (100.00%)
Posted the Same Day	19/22 (86.36%)
Posted the Same Week but not the Same Day	12/15 (80.00%)
Overall	102/240 (42.50%)

The survey data clearly indicates that certain characteristics of pairs of comments could predict the relationship level between them, which we could leverage with the aim of automating the determination of whether a pair of

Stack Overflow comments is (strongly) related, and should be displayed together to the reader. From this survey results, in the next section, we derive specific features based on the observed characteristics, e.g., same author, refers to author or temporal features. We use these features as part of a machine learning pipeline to automatically determine comment pair and comment group relatedness.

3 Methodology: Automatically Determining Comment Relatedness

In this section, we describe a set of features and machine learning algorithms that can be used to automatically determine if a pair of Stack Overflow comments is related.

Table 2 List of features used to determine the relatedness of a pair of Stack Overflow comments.

Feature	Type	Description
Author: same_speaker	binary	whether the pair of comments is by the same author or not.
Author: refers_to_speaker	binary	whether the author of one comment refers to the author of the second comment or not.
Author: refers_to_same_third_author	binary	whether the authors of the pair of comments refer to a common third author or not.
Author: refers_to_diff_third_author	binary	whether or not the authors of the pair of comments refer to two different authors.
Time: tdiff_5min	binary	if true, indicates that both comments in the pair were posted within five minutes.
Time: tdiff_hour	binary	the difference in the comments post time is between five minutes and 1 hour.
Time: tdiff_24h	binary	the difference in the comments post time is between 1 hour and 24 hours.
Time: tdiff_week	binary	the difference in the comments post time is between 24 hours and 1 week.
Time: tdiff_other	binary	the difference in the comments post time is above 1 week.
Text: jaccard	[0-1]	Jaccard similarity computed between the words occurring in the two comments.
Text: jaccard_code	[0-1]	Jaccard similarity computed between the words referring to code, occurring in the two comments.
Semantic: glove_cosine	[0-1]	Cosine similarity of the the average of the GloVe embeddings for each comment.
Semantic: weighted_glove_cosine	[0-1]	Cosine similarity of the weighted (by word occurrence frequency) average of the GloVe embeddings.

Features. Based on our analysis of the survey results and prior approaches to compute similarity among utterances in communication channels (e.g., [9]), we identified four groups of Stack Overflow comment similarity features: (1) *Author-based features* – concerned with the relationships between the authors of a pair of comments; (2) *Temporal features* – concerned with the elapsed time between a pair of comments; (3) *Text similarity features* – concerned with measuring the similarity in the terms in comment pair; (4) *Semantic*

features – concerned with establishing relationships in the semantics of the two comments. Table 2 lists the individual features within these four feature groups that we devised.

In order to compute the features, we use several different approaches. For the authorship-based features, same_speaker, refers_to_speaker, refers_to_same_third_author, and refers_to_diff_third_author, we used the Stack Exchange web API to retrieve authorship attributes for the Stack Overflow comments under our study. Since the Stack Exchange Web API imposes throttling and limits the number of daily calls, we cached the attribute data with a period refresh on need basis. We observed that it is quite common for the Stack Overflow users to change their display names, which are not propagated to the @author references in the comments. Therefore, we commonly encountered cases where a user referred to in a comment has changed her display name subsequently. This makes it difficult to establish a link between a pair of comments where one refers to the author of the other but the referee has changed her name. To address this problem, we make a call to Stack Exchange User API and retrieve all prior names of the authors of the comments under study and cache them for future use. Later at the time of evaluating referrer - referee relationship, if applicable, between a pair of comments, we compare the user name specified with @ symbol in a comment with all prior names of the author of the other comment.

In order to compute the temporal features, tdiff_5min, tdiff_hour, tdiff_24h, tdiff_week, and tdiff_other, we again relied on using the Stack Overflow API to retrieve the relevant timestamps (in UTC) and compute the difference.

For the jaccard text similarity feature, we tokenize each Stack Overflow comment and stem each token using the NLTK library [17]. We then compute the Jaccard similarity between the sets of tokens belonging to each comment. The jacard_code feature is computed similarly, but only considers the code tokens, which we define, simply, as those tokens that are in camelcase format.

The semantic features, glove_cosine and weighted_glove_cosine, are computed based on encoding the Stack Overflow comments' text with GloVe [20] word embeddings that we pretrained on the entirety of Stack Overflow (using the Stack Overflow data dump as of June, 2020) with default parameters (*vector size = 200; window size = 15*) [11]. We converted the Glove vectors to Word2Vec [18] vectors and used them with Gensim [23] Python library. For the glove_cosine feature we compute the $cosine\_similarity$ ($average(emb_{comment1})$, $average(emb_{comment2})$), where $emb_{comment}$ are the GloVe embeddings for an individual comment. The weighted_glove_cosine goes beyond this by taking the weighted average (using the smoothed inverse frequency) of the word embeddings in the sentence, i.e., every word embedding is weighted by $a/(a + freq(w))$, where a is a parameter that is set to 0.001 and $freq(w)$ is the estimated frequency of the word in the Stack Overflow data dump.

Relatedness Prediction. Our first goal is to automatically gauge the relatedness between a pair of Stack Overflow comments and, to this end, we use a traditional supervised learning approach based on the features described above. We initially frame the problem as binary classification, with a positive

output indicating that a pair of comments is related and should be displayed together, and a negative output indicating that they are unrelated. We attempted several popular machine learning algorithms, e.g., Random Forest, K-Nearest-Neighbors, Artificial Neural Network and Support Vector Machine, to train this model, relying on their implementations in the scikit-learn library. We performed informal hyperparameter tuning of the algorithms, e.g., experimenting with settings such as the number of trees and the maximum tree depth for Random Forest. Observing no significant improvement in performance, we opted to rely on default settings for each of the hyperparameters. In examining the different classification algorithms, we observed that Random Forest ($number\_of\_estimators = 1,500; max\_depth = 5$) performed slightly better than the others. This algorithm also has the ability to produce binary as well as probabilistic output (between 0 and 1) by computing the mean predicted class probabilities of the trees in the Random Forest. Since the dataset is class imbalanced, i.e., the number of unrelated comment pairs exceeds the related ones, we experimented with oversampling using SMOTE. We found no significant improvement in the classification accuracy when using SMOTE, likely due to the fact that the imbalance ratio was not large (the unrelated comments outnumbered the related ones 2:1).

Clustering Comments. Our second goal was in producing groups (or clusters) of related comments from the set that surround a single Stack Overflow question or answer. For this purpose, as input we used the prediction probability produced by the Random Forest algorithm as detailed above. The prediction probabilities form the weighted edges of a graph, where the nodes are the individual Stack Overflow comments. We apply a graph cutting algorithm to divide this graph into groups of nodes with high relatedness between them (i.e., clusters of comments). We leveraged a greedy voting approach proposed by Elsner and Charniak for chat disentanglement [9]. More specifically, this algorithm considers whether to assign a Stack Overflow comment j to a cluster C by examining all previously assigned comments C_i in that cluster and treating the classifier's prediction weight, $w_{ij} >= 0.5$, as a vote for j. If the overall vote is greater than 0, j in put in C, and otherwise j is put in a newly created cluster. The algorithm operates by simply processing each Stack Overflow comment, one by one, in ascending order of their posting time. The ability to operate efficiently online is a considerable advantage of this algorithm, as the problem of graph cutting is computationally complex due to the potential for transitive relationships among the comments, e.g., comment A is related to comment B, comment B is related to comment C, comment C is related to comment A.

4 Evaluation Plan

In this section, we describe how we curate a dataset and select metrics in order to evaluate our approach for automatically determining Stack Overflow comment relatedness. Specifically, we evaluate for two related purposes: 1)

Fig. 6 Conversation groups formed by the comments on a Stack Overflow answer/

automatically determining relatedness of a pair of comments (in order to build comment recommendation system); and 2) automatically identifying groups of related comments (that can be displayed in lieu of how currently Stack Overflow's interface shows comments).

Dataset. To curate a suitable dataset for evaluation, we began by selecting 50 answers from Stack Overflow that have more than 5 comments each. This is because the Stack Overflow comment hiding mechanism is only applied to answers that have more than 5 comments. To select high quality Stack Overflow answers on a popular topic, we performed a search on Stack Overflow with the keyword 'java' and sorted the results by number of votes in descending order. The idea behind sorting was that the more votes an answer has, the higher its quality and popularity and hence a higher chance of having answers with multiple comments. We selected 50 such answers having more than five comments each. The number of comments on the answers we sampled varied broadly, from 6 to as many as 85. In total, our dataset consists of 871 Stack Overflow comments and 11,822 comment pairs.

The comments on a given Stack Overflow answer may form one or more separate conversation threads, while each comment belongs to exactly one of the conversation threads. For example, consider the answer with 9 comments shown in Figure 6. As shown, the comments 1 and 2 form a conversation group 1 where the users discuss whether the Python list concatenation method proposed in the answer is a deep copy or a shallow copy. The comments 3, 4, 5, 6 and 9 discuss a tweak in the original answer using a "+=" operator to concatenate lists. Finally, the comments 7 and 8 discuss the concatenation of *numpy* arrays. Thus, in this example, the comments form 3 conversation groups.

The two authors reviewed the comments for each of the 50 answers (total of 871 comments) with a goal of identifying their conversation groups. The grouping also implicitly establishes related pairs of comments, which we also needed for part of our evaluation strategy. Each author annotated the comments separately, taking notes to register the rationale for more nuanced decisions. Following this, the authors compared their annotations, discussing and resolving differences in the individual annotations to create a final annotated evaluation dataset.

Metrics. For the relatedness of a pair of comments we chose standard metrics used for binary classification problems, i.e., precision, recall and the micro-averaged F-score,

$$F\text{-}score = 2 * \frac{precision * recall}{precision + recall} = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

In the micro-averaged case, the TP, FP and FN are totalled across both classes, which is appropriate for imbalanced data.

For comment clustering, we selected metrics that were previously used for similar problems like chat disentanglement: one-to-one accuracy and conversational micro-average F-score. The one-to-one accuracy is computed as the percentage overlap when conversations from two annotations are optimally paired up [9]. Different from the comment-pair F-score, which is based on binary classification, for clustering we consider micro-averaged F-score where each cluster is considered as a separate class. To address the fact that the data is imbalanced, we use ROC AUC, the area under the ROC (Receiver Operating Characteristics) curve, which represents the degree of separability between prediction classes. The value of the ROC AUC metric ranges between 0 and 1, where 1 represents complete agreement between the prediction and the gold set.

5 Results

We detail the evaluation results for pairwise comment relatedness and comment clustering, in turn.

5.1 Comment Relatedness

To examine the performance of the model and the effect of different feature types, we selected random and ablation baselines. More specifically, the *Random* baseline randomly chooses whether two comments are related, while the *Semantic features*, *Text Similarity features*, *Speaker features*, and *Temporal features* uses the same machine learning algorithm and configuration as the complete model (i.e., *All features*) but only a subset of the features. Table 3 summarizes the performance of the model and the baselines. To compute the

Table 3 Evaluation results of our pairwise comment relatedness model.

Technique	**Precision**	**Recall**	**F-score**	**ROC AUC**
Random	0.416	0.486	0.498	0.487
Semantic features	0.478	0.035	0.559	0.489
Text Similarity features	0.390	0.031	0.579	0.506
Speaker features	0.742	0.675	0.749	0.739
Temporal features	0.762	0.680	0.766	0.755
All features	0.802	0.684	0.779	0.767

metrics for each configuration, we used 10-fold cross validation, applying it separately for each of metrics.

For pairwise relatedness, our model including all the features revealed an average ten-fold cross validation precision of 0.802, recall of 0.684, F-score of 0.779, and ROC AUC of 0.767, indicative of the model being useful for the purpose it was designed for. Table 3 reflects the performance for the Random Forest classifier, while the other classifiers that we experimented with (K-Nearest Neighbors, Artificial Neural Network, Support Vector Machine, XGBoost and LightGBM) all performed slightly worse, e.g., with a lower F-score ranging from 0.002 (Support Vector Machine) to 0.075 (K-Nearest Neighbors), 0.027 (XGBoost) and 0.011 (LightGBM).

While the model using all four features sets performs better than all baselines, it is clear that some feature sets perform poorly (i.e., *Text Similarity features* and *Semantic features*) while the others (i.e., *Speaker features* and *Temporal features*) perform nearly as well as the combination of all features. The discrepancy among the performance of the feature sets towards comment relatedness prediction reveals a few observations. The feature sets that were particularly effective, based on temporal and speaker characteristics, were the simpler to compute, but indicate that those simple characteristics can be used effectively to predict comment relatedness. On the other hand, the features based on text and semantics, while key in chat disentanglement [9, 15], were largely ineffective here because of the narrow topic of the Stack Overflow comments, i.e., all of the comments are discussing the very same software development concepts and programming elements so this is not a good differentiating factor.

5.2 Comment Clustering

To contrast the performance of our approach towards comment clustering, as baseline we use the current approach used by Stack Overflow to group and display comments, which is based on comment vote totals. Based on this information, we create two clusters, one with all of the (top-voted) visible comments and another with all the remaining comments. The results of our clustering and the baseline are shown in Table 4. We use a lower voting algorithm threshold of $w_{ij} >= 0.4$ for a positive vote, instead of the default of

Table 4 Evaluation results of our comment clustering model.

Technique	One-to-one Accuracy	F-score
Stack Overflow Vote-Based Grouping	0.436	0.534
Our Clustering	0.516	0.613

0.5, after observing that the distribution of comment similarity weights have a slightly lower mean, which would have resulted in the creation of many very small clusters.

The clustering results in Table 4 lead to a few observations. First, our clustering technique provides significant improvements over the current default comment display strategy used by Stack Overflow. However, this is probably to be expected as Stack Overflow, in its comment display strategy emphasizes individual highly voted comments instead of clusters or groups.

Second, we observe that while the relatedness among pairs of comments can be leveraged to improve the selection of the related groups of comments, the absolute values of the two metrics, F-score and One-to-one Accuracy, are relatively low. We believe that to improve the quality of the clustering, we must first improve the comment pairwise relatedness since the clustering is driven by the strength of the relationship between pairs of comments. However, as we observed above, Stack Overflow comment relatedness is difficult due to the narrow domain of the comments, i.e., all the comments are centered around a single Stack Overflow question or answer.

6 Threats to Validity

A threat to construct validity arises from the annotators misclassifying the relatedness of Stack Overflow comments. To limit this threat, we selected annotators with prior experience in software development. We also computed Cohen's Kappa between the pairs of annotators that annotated the same comments, observing a value of 0.4 that can be interpreted as fair agreement. A similar threat arises for the gold set, which was annotated by the two authors. We mitigated this threat by having both authors annotate the entire dataset and then carefully and systematically resolving differences.

The results of our comment relatedness study also suffer from internal validity due to our process for selecting comment pairs using seed utterances, which may bias the selection towards more related or less representative pairs of utterances. We used this selection strategy in order to focus on Stack Overflow comments that are typically hidden by the default behavior of the platform, which is the problem we set out to address with this work. In addition, using seed utterances and previous and successive instances is an approach previously used by researchers analyzing similar data [25].

The reported results leverage machine learning algorithms (e.g., Random Forest, XGBoost and LightGBM) with hyperparameters that were not tuned via a systematic parameter search. A threat to the internal validity is that

we did not use optimal parameter choices, which may strongly influence the study results. A mitigating factor is that we informally experimented with different hyperparameters for the classifiers and observed little sensitivity when choosing reasonable parameter values.

While the gold set consists of a reasonable number of comments (871), these correspond to only 50 Stack Overflow posts. This limited number of posts creates a threat to external validity as the selected posts may not be representative of typical Stack Overflow posts. We mitigated this threat by selecting popular Stack Overflow posts, according to the number of votes received on the platform, which are likely to be of high quality.

7 Related Work

The related work comes from three different topics. First and most related to this paper, researchers have written about directly addressing the problem of Stack Overflow comment hiding. Second, researchers have considered how Stack Overflow comments can be leveraged for building automated tools or performing empirical analyses of developer behavior. Lastly, approaches that cluster or compute relatedness between conversational utterances from multiple participants (e.g., chats) are also related to this work, so we outline a few notable techniques.

Stack Overflow comment hiding. Rahman, et al. [22] proposed a Stack Overflow comment mining technique leveraging Latent Dirichlet Allocation (LDA) topic modeling and the *pagerank* algorithm in order to identify insightful comments for answers containing source code. They also performed an exploratory study where they analyzed Stack Overflow follow-up discussions, and reported that the comments contained useful information for static analysis and also identified issues, deficiencies and scopes for further improvement in the code. Zhang et al. performed an empirical study on the comment hiding mechanism of Stack Overflow [30] and observed that about 4.4 million comments are hidden. Furthermore, by analyzing 1.3 million answers that have hidden comments Zhang et al. observed the following. 1) Hidden comments are as informative as displayed comments; more than half of the comments (both hidden and displayed) are informative. 2) The current comment hiding mechanism tends to rank and hide comments based on their creation time instead of their score in most cases due to the large amount of tie-scored comments. This is also because 87% of the comments on Stack Overflow have a score of 0, that is, no upvotes. 3) In 97.3% of answers that have hidden comments, at least one comment is hidden while there is another comment with the same score is displayed. The authors recommend that Stack Overflow should consider adjusting their current comment hiding mechanism and that users examine all comments, in case they would miss informative details such as software obsolescence, code error reports, or notices of security vulnerabilities. As a follow on to this analysis, Zhang et al. [31] built a classifier that could effectively distinguish informative comments from uninformative Stack Overflow comments.

They evaluated two alternative comment organization mechanisms based on text relatedness.

Motivated by this research, in this paper, we explore the idea that Stack Overflow comments contain useful knowledge and that they should not be read in isolation, but that often comments form a conversation, and therefore it is important to understand the conversational context to understand a specific Stack Overflow comment. We evaluate this hypothesis using an exploratory survey revolving around Stack Overflow comments.

Leveraging Stack Overflow comments. Sengupta, et al [24] conducted a content analysis on Stack Overflow comments, identifying nine categories of comments, e.g., comments expressing affect, comments with improvements. The categories suggest that Stack Overflow comments are of significant value to the Stack Overflow discussions and to the Stack Overflow community, and contain potentially valuable information that can be leveraged by other sites or tools. Zhang et al. [29] investigated 32.3 million comments that were associated with answers on Stack Overflow, revealing that 23% (i.e., 2.6 million) of the answers with comments have a commenting thread longer than the actual answer, indicating the richness of information in comments. They found that the majority of comments are informative as they enhance answers with a more diverse range of perspectives. They also find that comments are rarely integrated into their associated answers, indicating that reading answers alone on Stack Overflow is not as informative. Diyanati, et al [8] performed studies to determine Stack Overflow users' expertise level based on two methods. While the emphasis of the first method was on the scores of the question and answers of the users, they found that these had no significant relationship with the user's expertise as determined by the user's reputation score. On the other hand, their second method showed that positive and negative comments provided by other users on questions or responses of a specific user can be used to identify the level of expertise of each user. In an investigative study on the importance of comments in a large-scale programming forums such as Stack Overflow, Aggarwal, et al. [6] randomly sampled 22K Stack Overflow comments on 5025 questions and 3013 answers. They extracted syntactic, semantic and social features of the posts and the comments and conducted a regression analysis with post and comment features to model the quality of the post as measured by the number of votes. Their study confirmed that comment semantics not only shape the answer quality but also increase the overall model explanatory power. Novielli, et al performed annotation of text containing emotions on Stack Overflow, chiefly focusing on comments. They created a gold set corpus [19] consisting of 4,800 data elements annotated with one or more of six emotion categories. Calefato, et al. developed an emotion recognition toolkit named EmoTxt [7] that they trained on corpora that included the Stack Overflow comments goldset developed by Novielli, et al.

In this paper, we consider a different angle in examining how the relationship between pairs and groups of comments can be used to understand what comments are most relevant to readers. Our experimental results show that

automatic ML-based approaches can help determine comment relatedness, but that there is more work to be done for doing this with high accuracy.

Utterance similarity and clustering. Kummerfeld, et al [15] tackled the problem of (chat) conversation disentanglement, which aims to create salient conversations from interleaved utterances on a chat channel. They introduced a large-scale conversation disentanglement dataset of 77,563 messages mined from IRC, which they manually annotated with reply-to relations between messages. Stack Overflow comments also exhibit similar characteristics to IRC chats (e.g., multiple interleaved utterances by different users) and are therefore could be a good candidate for a such technique. However, unlike chat channels, Stack Overflow messages are focused on a much more narrow topic, i.e., a single Stack Overflow answer. Jiang, et al [14] used a Siamese Hierarchical Convolutional Neural Network (SHCNN) and Conversation Identification by Similarity Ranking (CISIR) techniques to conduct experiments on Reddit and IRC conversation datasets for performing conversation disentanglement. Such datasets and methods can be used to develop robust data-driven methods for conversation disentanglement task on a single stream of interleaved messages such as Stack Overflow comments on an answer. Elsner, et al [9] demonstrated a method which employed key characteristics of the conversations like speaker, time gap, mention and content based features to train a classifier for performing chat disentanglement on an IRC dataset.

Through this research, we observe that, compared to chat channels, Stack Overflow messages are focused on a much narrower topic - a single Stack Overflow answer. In this paper, we propose a machine learning based method to automatically identify the relatedness of a pair of Stack Overflow comments. Combining comment relatedness with the chat disentanglement algorithm of Elsner et al. [9], we identify an approach that forms clusters of related comments on a specific Stack Overflow post, which can aid readers in getting the full conversational context of a comment.

8 Conclusion And Future Work

In this paper, building on prior work that showed that the Stack Overflow's comment hiding mechanism is not effective, we examined automatic approaches for determining comment relatedness and for grouping comments. We posit that since comments form distinct conversational threads, showing the entire cluster of related comments instead of individual ones is more effective in order to understand the context of the discussion. We identified key features and demonstrated a machine learning-based method to identify relatedness of a pair of comments in an automatic way. We further described a graph cutting algorithm that formed a cluster of similar comments based on this machine learning model. Both the comment relatedness and clustering showed performance that improves upon the selected baselines.

The future work of this project includes investigating machine learning models that are more effective towards comment relatedness detection and

clustering by examining additional features and algorithms. A promising direction is applying the latest generation of chat disentanglement algorithms that have shown strong improvements in solving this problem by using, e.g., graph neural networks. Another avenue of future work is in studying how our techniques can be best integrated with the current Stack Overflow user interface in order to hide or display the clusters of similar comments.

References

1. Comment hiding mechanism on Stack Overflow. `https://meta.stackexchange.com/a/17365/744448`. Accessed: 2021-06-29
2. Stack Exchange. `https://www.stackexchange.com`. Accessed: 2021-06-29
3. Stack Overflow number of comments as of January 2021. `https://data.stackexchange.com/stackoverflow/query/1364273`. Accessed: 2021-06-29
4. Stack Overflow public questions and answers. `https://www.stackoverflow.com`. Accessed: 2021-06-29
5. Stack Overflow statistics as of November 2020. `https://stackexchange.com/sites?view=list#users`. Accessed: 2021-06-29
6. Aggarwal, A., López, C., Hsiao, I.H.: The role of comments' controversy in large-scale online discussion forums. In: Proceedings of the 27th ACM Conference on Hypertext and Social Media, pp. 179–182 (2016)
7. Calefato, F., Lanubile, F., Novielli, N.: Emotxt: a toolkit for emotion recognition from text. In: 2017 seventh international conference on Affective Computing and Intelligent Interaction Workshops and Demos (ACIIW), pp. 79–80. IEEE (2017)
8. Diyanati, A., Sheykhahmadloo, B.S., Fakhrahmad, S.M., Sadredini, M.H., Diyanati, M.H.: A proposed approach to determining expertise level of Stack Overflow programmers based on mining of user comments. Journal of Computer Languages **61**, 101000 (2020)
9. Elsner, M., Charniak, E.: Disentangling chat. Computational Linguistics **36**(3), 389–409 (2010)
10. Ericson, J.: Wealth condensation resulting from top n comments system. `https://meta.stackexchange.com/a/204579/744448`. Accessed: 2021-06-29
11. Imran, M.M., Ciborowska, A., Damevski, K.: Automatically selecting follow-up questions for deficient bug reports. In: Proceedings of the 18th International Conference on Mining Software Repositories (MSR'21) (2021)
12. Jaydles: Hidden comments contain valueable information. `https://meta.stackexchange.com/a/209808/744448`. Accessed: 2021-06-29
13. Jaydles: Purpose of users commenting on Stack Overflow questions and answers. `https://meta.stackexchange.com/a/209808/744448`. Accessed: 2021-06-29
14. Jiang, J.Y., Chen, F., Chen, Y.Y., Wang, W.: Learning to disentangle interleaved conversational threads with a siamese hierarchical network and similarity ranking. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pp. 1812–1822 (2018)
15. Kummerfeld, J.K., Gouravajhala, S.R., Peper, J., Athreya, V., Gunasekara, C., Ganhotra, J., Patel, S.S., Polymenakos, L., Lasecki, W.S.: A large-scale corpus for conversation disentanglement. arXiv preprint arXiv:1810.11118 (2018)
16. Landis, J.R., Koch, G.G.: The measurement of observer agreement for categorical data. biometrics pp. 159–174 (1977)
17. Loper, E., Bird, S.: Nltk: The natural language toolkit. arXiv preprint cs/0205028 (2002)
18. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
19. Novielli, N., Calefato, F., Lanubile, F.: A gold standard for emotion annotation in Stack Overflow. In: 2018 IEEE/ACM 15th International Conference on Mining Software Repositories (MSR), pp. 14–17. IEEE (2018)

20. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1532–1543 (2014)
21. Ponzanelli, L., Mocci, A., Bacchelli, A., Lanza, M., Fullerton, D.: Improving low quality Stack Overflow post detection. In: 2014 IEEE International Conference on Software Maintenance and Evolution, pp. 541–544 (2014). DOI 10.1109/ICSME.2014.90
22. Rahman, M.M., Roy, C.K., Keivanloo, I.: Recommending insightful comments for source code using crowdsourced knowledge. In: 2015 IEEE 15th International Working Conference on Source Code Analysis and Manipulation (SCAM), pp. 81–90. IEEE (2015)
23. Řehůřek, R., Sojka, P.: Software Framework for Topic Modelling with Large Corpora. In: Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks, pp. 45–50. ELRA, Valletta, Malta (2010). http://is.muni.cz/publication/884893/en
24. Sengupta, S., Haythornthwaite, C.: Learning with comments: An analysis of comments and community on Stack Overflow. In: Proceedings of the 53rd Hawaii International Conference on System Sciences (2020)
25. Shi, L., Chen, X., Yang, Y., Jiang, H., Jiang, Z., Niu, N., Wang, Q.: A first look at developers' live chat on gitter. In: Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE 2021, p. 391–403. Association for Computing Machinery, New York, NY, USA (2021). DOI 10.1145/3468264.3468562. URL https://doi.org/10.1145/3468264.3468562
26. Stocker, G.: Comment Explosion. https://meta.stackexchange.com/q/180325/744448. Accessed: 2021-06-29
27. user200500: Asker's comments getting no votes while answer's comments getting more upvotes. https://meta.stackexchange.com/questions/204402/hide-trivial-comments#comment652961_204402. Accessed: 2021-06-29
28. Yao, Y., Tong, H., Xu, F., Lu, J.: Scalable algorithms for cqa post voting prediction. IEEE Transactions on Knowledge and Data Engineering **29**(8), 1723–1736 (2017). DOI 10.1109/TKDE.2017.2696535
29. Zhang, H., Wang, S., Chen, T.H., Hassan, A.E.: Reading answers on Stack Overflow: Not enough! IEEE Transactions on Software Engineering (2019)
30. Zhang, H., Wang, S., Chen, T.H.P., Hassan, A.E.: Does the hiding mechanism for Stack Overflow comments work well? no! arXiv preprint arXiv:1904.00946 (2019)
31. Zhang, H., Wang, S., Chen, T.H.P., Hassan, A.E.: Are comments on Stack Overflow well organized for easy retrieval by developers? ACM Transactions on Software Engineering and Methodology **30**(2) (2021). DOI 10.1145/3434279. URL https://doi.org/10.1145/3434279