Are Comments on Stack Overflow Well Organized for Easy Retrieval by Developers?

HAOXIANG ZHANG, Centre for Software Excellence, Huawei, Canada
SHAOWEI WANG, Department of Computer Science, University of Manitoba, Canada
TSE-HSUN (PETER) CHEN, Software PErformance, Analysis, and Reliability (SPEAR) Lab, Concordia University, Canada
AHMED E. HASSAN, Software Analysis and Intelligence Lab (SAIL), Queen's University, Canada

Many Stack Overflow answers have associated informative comments that can strengthen them and assist developers. A prior study found that comments can provide additional information to point out issues in their associated answer, such as the obsolescence of an answer. By showing more informative comments (e.g., the ones with higher scores) and hiding less informative ones, developers can more effectively retrieve information from the comments that are associated with an answer. Currently, Stack Overflow prioritizes the display of comments, and, as a result, 4.4 million comments (possibly including informative comments) are hidden by default from developers. In this study, we investigate whether this mechanism effectively organizes informative comments. We find that (1) the current comment organization mechanism does not work well due to the large amount of tie-scored comments (e.g., 87% of the comments have 0-score) and (2) in 97.3% of answers with hidden comments, at least one comment that is possibly informative is hidden while another comment with the same score is shown (i.e., unfairly hidden comments). The longest unfairly hidden comment is more likely to be informative than the shortest one. Our findings highlight that Stack Overflow should consider adjusting the comment organization mechanism to help developers effectively retrieve informative comments. Furthermore, we build a classifier that can effectively distinguish informative comments from uninformative comments. We also evaluate two alternative comment organization mechanisms (i.e., the *Length* mechanism and the *Random* mechanism) based on text similarity and the prediction of our classifier.

CCS Concepts: • **Software and its engineering → Maintaining software**; **Documentation**;

Additional Key Words and Phrases: Empirical software engineering, crowdsourced knowledge sharing, Q&A website, stack overflow, commenting

ACM Reference format:
Haoxiang Zhang, Shaowei Wang, Tse-Hsun (Peter) Chen, and Ahmed E. Hassan. 2021. Are Comments on Stack Overflow Well Organized for Easy Retrieval by Developers? *ACM Trans. Softw. Eng. Methodol.* 30, 2, Article 22 (February 2021), 31 pages.
https://doi.org/10.1145/3434279

Authors' addresses: H. Zhang, Centre for Software Excellence, Huawei, Canada; email: hzhang@cs.queensu.ca; A. E. Hassan, Software Analysis and Intelligence Lab (SAIL), Queen's University, Kingston, Ontario, Canada; email: ahmed@cs.queensu.ca; S. Wang, Department of Computer Science, University of Manitoba, Winnipeg, Manitoba, Canada; email: shaowei@cs.umanitoba.ca; T.-H. (Peter) Chen, Software PErformance, Analysis, and Reliability (SPEAR) Lab, Concordia University, Montreal, Quebec, Canada; email: peterc@encs.concordia.ca.

1049-331X/2021/02-ART22 $15.00
https://doi.org/10.1145/3434279

1 INTRODUCTION

Stack Overflow is a Q&A platform that is widely used by software developers to find answers to their programming questions. It has over 50 million visitors in a single day.[1] In the archived data provided by Stack Overflow in September 2017, there are 22.7 million answers to questions in various domains related to software development. These answers offer developers solutions to address their questions. This collection of answers enables developers to learn and share valuable programming knowledge.

An answer creates a starting point for a discussion related to a question. Users can post alternative answers, edit existing answers, or post comments under the answer. We consider an answer thread consisting of an answer and all the associated comments. Comments can provide additional information to support their associated answer, or even point out issues in the answer, such as the obsolescence of answers [73]. Such information can be very helpful to developers.

However, the more comments that are posted under an answer, especially those answers that attract large user traffic, the more needed effort and time that developers need to retrieve information on Stack Overflow. To keep each answer thread compact, Stack Overflow implements a comment organization mechanism to only show the top 5 comments. Aiming at showing the most informative comments and hiding less informative ones, the mechanism first ranks these comments based on their scores. When multiple comments have the same score, they are then ranked by their creation time. In addition, informative comments might be hidden by the comment organization mechanism in turn reducing the chances of someone voting on them. To read the hidden comments, users need to click a link under the last shown comments. In addition, hidden comments are not indexed by Google,[2] which also hinders the accessibility to the information in comments for answer seekers.

We observe that 4.4 million comments are hidden as of September 2017, and as a result a large amount of useful information is probably buried in such hidden comments. As once commented by David Fullerton (the president of Stack Overflow) in a discussion of "hide trivial comments" on Stack Exchange META, "*if I have to click that link every time just in case there's something useful in the comments, haven't we failed?*"[3] In other words, it is essential for Stack Overflow to show the most useful comments and hide less useful ones to ease the information retrieval for developers. Therefore, the decision of how to organize Stack Overflow comments is essentially an information retrieval (IR) task within a Software Engineering context—how to select the most informative comment to display to developers in a non ad hoc manner. Selecting informative comments from answers is similar to prior IR and software engineering (SE) research efforts, such as bug localization [22, 52, 61] and source code recommendation [31], which aim to assist developers by effectively retrieving information from large and unorganized corpuses. The organization of comments can negatively impact the effective retrieval of useful comments, since the ordering of comments leads to some comments to be hidden from developers and search engines (which is a major problem, since many developers reach Stack Overflow answers through search engine results [1, 2, 67, 70]).

Therefore, it is important to understand how well the comment organization mechanism works. Does this mechanism, in fact, show the more informative comments and hide less informative comments as designed? What types of information are actually discussed in hidden and shown comments? By answering these questions, we explore and raise the importance of a proper comment organization mechanism and wish to provide insights to improve the current mechanism to make it easier for developers to retrieve information on Stack Overflow.

[1] Data obtained on Jan 31, 2018, from https://insights.stackoverflow.com/survey/2018/.
[2] https://meta.stackexchange.com/a/304906/.
[3] https://meta.stackexchange.com/posts/comments/653443/.

In this article, we study 22.7 million answers and all of their 32.2 million associated comments. We first qualitatively study whether the shown comments are more informative than the hidden comments. In other words, is the comment organization mechanism actually hiding less informative comments? We investigate the following.

- **RQ1: What are the characteristics of both hidden and shown comments?**
  We observe that hidden comments have a similar amount of text as shown comments. Hidden comments share less semantic similarity with the associated answers than that of shown comments in the majority of the answers with hidden comments. Based on our qualitative study, we observe that more than 70% of the comments (both hidden and shown) are informative, as they provide alternative answers, or point out flaws in answers.

From the previous qualitative study, we observe that many informative comments can be hidden due to the current comment organization mechanism. To further understand the reason for such cases, we perform an empirical analysis to evaluate the efficacy of the comment organization mechanism by answering the following two RQs:

- **RQ2: How effective is the current comment organization mechanism?**
  The comment organization mechanism does not work effectively. Instead of giving priority to highly scored comments, it gives priority to early comments, since the comment organization mechanism fails to consider a very common case: Multiple comments may have the same score (i.e., tie-scored comments). More specifically, in 97.3% of the answers that have hidden comments, at least one comment is hidden (i.e., unfairly hidden comment), while other comments with the same score are shown (i.e., unfairly shown comments).
- **RQ3: What are the characteristics of unfairly hidden comments?**
  Based on our qualitative study, we obverse that the longest unfairly hidden comments are more likely to be informative than the shortest unfairly shown comments (when the shortest unfairly shown comments are less than 50 characters).

Based on above findings, we suggest that Stack Overflow enhances its comment organization mechanism to better handle tie-scored comments that represent 92.9% of the hidden comments. Useful information can be embedded in these hidden comments. Developers could retrieve such information more effectively if hidden comments are more properly organized. Instead of simply ranking comments by their score then their creation time, the comment organization mechanism needs to introduce a higher priority for more informative comments, which can ease the retrieval of information for developers. For example, Stack Overflow can replace the shortest unfairly shown comments with the longest unfairly hidden comments (i.e., with tied score). Such a new mechanism will ensure that such informative comments are indexed by search engines. To evaluate our findings, we build a classifier (with an AUC of 0.8) to distinguish informative comments from uninformative comments. We evaluate two alternative comment organization mechanisms (i.e., the *Length* mechanism and the *Random* mechanism) based on text similarity and the prediction of our classifier. Our analysis shows that the *Random* mechanism is an option to diversify the comments, and the *Length* mechanism enables an improvement to show informative comments. In addition, we encourage developers to read through all comments (including hidden comments) in case any further correction/improvement is made by such comments, such as observations of answer obsolescence, security vulnerabilities, and errors.

Paper Organization: The rest of the article is organized as follows. Section 2 introduces background about Stack Overflow's comment organization mechanism. Section 3 details the dataset that we used in this study. Section 4 and Section 5 present the case study results. Section 6 discusses

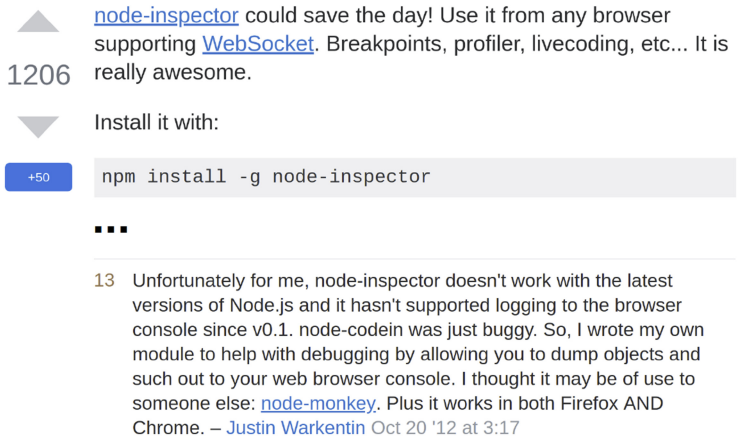

Fig. 1. An answer with one of its associated comments. The comment notes that the node-inspector no longer works with the recent versions of Node.js and proposes the use of an alternative module called node-monkey.

our findings and provides actionable suggestions. Section 7 discusses potential threats from our case study. Section 8 surveys prior work related to our study. Finally, Section 9 concludes our study.

2 BACKGROUND

2.1 Commenting on Stack Overflow

Commenting on Q&A websites can lead to more comprehensive discussions that improves the knowledge sharing process [19, 42, 73]. For example, as shown in Figure 1, a user on Stack Overflow posted a comment[4] that pointed out a problem with an existing answer and provided an alternative solution. On Stack Overflow, comments are "temporary 'Post-It' notes left on a question or answer"[5] *As of September 2017, users on Stack Overflow have posted 60.2 million comments* under questions or answers, which represents more than the total number of questions and answers combined according to the data from Stack Exchange Data Explorer.[6] Note that in this study we refer to comments as *the comments that are associated with answers if not specified otherwise*, since we focus on studying how comments contribute to providing additional value to answers. Even though new answers and revisions that contribute to existing answers can also provide additional values to existing answers, our study focuses on investigating whether comments are organized properly.

Comments open up a channel for developers to add informative discussions to their associated answers. Highly informative comments can expedite the problem solving process or add knowledge that is worthwhile to share for a particular answer, such as the observation or update of an obsolete answer [73]. As explained by Jeff Atwood (the co-founder of Stack Overflow), "*there are often important clarifications and addendums left as comments that substantially improve the original post*" [50]. In addition, Chang et al. [11] note that comments are a *first class citizen* of a Q&A platform. They can critically improve an answer by providing additional clarifications and refinements to the answer; thus, increasing the overall value of an answer. In Section 4.1.2, we also observe that comments provide value for various aspects of an answer, such as pointing out errors, making corrections, and providing alternative solutions. Therefore, commenting is

[4] https://stackoverflow.com/posts/comments/17612489.
[5] https://stackoverflow.com/help/privileges/comment.
[6] https://data.stackexchange.com/stackoverflow/query/945995.

an indispensable Stack Overflow feature that leads to improving and ensuring the long-lasting value of its crowdsourced knowledge. In other words, such informative comments can provide additional knowledge to developers when they seek answers.

However, uninformative comments could potentially threaten the Stack Overflow community by decreasing the information density of an answer thread (i.e., the compactness of an interface in terms of the amount of information) [27]. If an increasing number of answer threads are filled up with discussions, such as through uninformative comments, then developers may face difficulties in identifying relevant information.

Therefore, to promote informative comments and avoid uninformative ones, several sorting rules are applied to comments on Stack Overflow. First, comments can only be posted by the following three types of users: the asker, the answerer, and any user with at least 50 reputation points.[7] Of the users, 11.7% have at least 50 reputation points on Stack Overflow. On Stack Overflow, reputation points can be earned by various approaches, for example, 10 points if a question is upvoted, 10 points if an answer is upvoted, 15 points if an answer is marked accepted, and 2 points if a suggested edit is accepted.[8] Stack Overflow also incorporates a voting system to regulate the quality of comments.[9] *Comments can be up-voted, but cannot be down-voted. Thus, the lowest score a comment can have is 0.*

In short, commenting is widely used to structure discussions around their associated answers. In addition, comments can be as informative as their associated answers to some extent. Thus, in this article, we study what developers actually discuss in comments and characterize the informativeness of comments.

2.2 The Organization of Comments on Stack Overflow

Initially, Stack Overflow used to hide all comments under answers; however, this rule was abandoned, because it hid too much information. As explained by Jeff Atwood that "comments were all locked behind ... information was being lost" [50]. However, developers may find it difficult to locate useful information if all comments were shown under each answer. The single webpage that contains question, answers, and comments would have an increasing amount of content over time. Hence, developers will need to spend more effort and time to read and locate the relevant information.

To have a better balance between showing and hiding all comments, Stack Overflow implemented its current comment organization mechanism to enhance the readability of answer threads. Since 2009 (Stack Overflow was launched in 2008), this comment organization mechanism only shows the "top 5 comments" [50] for each answer (i.e., *shown comments*). Note that if two comments have the same score (i.e., tied score), they will be ranked by their creation time and the earlier created comment will be ranked higher. Additionally, if more than 30 answers are posted for a question, then only the comments with a score greater than 0 will be shown. For example, if a question has more than 30 answers and one associated answer only has three comments that have a score greater than 0, then only these three comments will be shown under the answer. Other comments under an answer are hidden (i.e., *hidden comments*). Even comments that have a score that is greater than 0 will be hidden if there are more than 5 comments with a score greater than 0 under their associated answer (with 24,438 such cases, that is, 1.9% of the answers that have hidden comments), since only the top 5 comments are shown.

[7] https://stackoverflow.com/help/privileges/comment.
[8] https://stackoverflow.com/help/whats-reputation.
[9] https://meta.stackexchange.com/a/17365.

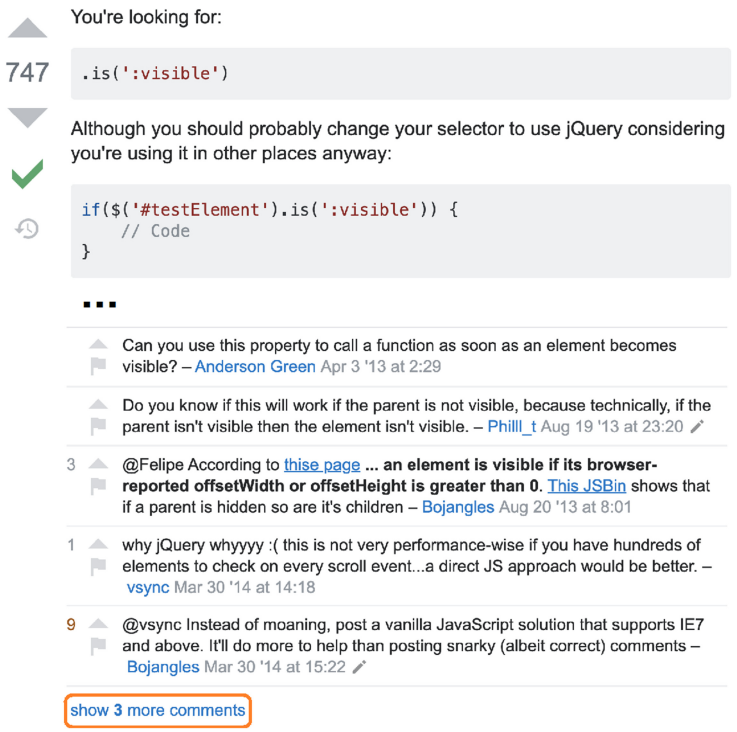

Fig. 2. An example of an answer with its shown comments and a link at bottom saying "show 3 more comments." Once a user clicks the link, all comments (including the hidden comments) under the answer will be shown.

Users can click a link saying "show n more comments" to read all comments. An example[10] of an answer thread is shown in Figure 2 with the shown comments for the answer and the clickable link at the bottom of the comments. By using the comment organization mechanism, ideally the three hidden comments are less likely to add additional information to their associated answers, while all the informative comments are shown. In this study, we investigate this comment organization mechanism to understand how it enables effective information retrieval on Stack Overflow.

The comment organization mechanism has been in place for nearly 10 years (as of October 2019) while the Stack Overflow community has expanded significantly since the launch of Stack Overflow. The number of users and answers have increased considerably. The number of comments also grows over the years [72]. However, the effectiveness of Stack Overflow's comment organization mechanism in showing informative comments and hiding uninformative comments remains unknown. Therefore, we wish to study whether comments (including both hidden and shown comments) are informative, and we investigate the efficacy of the comment organization mechanism. Moreover, it is common that developers use search engines to look for solutions to their questions, and many top search results are from Stack Overflow answers [1, 2, 67, 70]. However, hidden comments are not indexed by search engines[11] [50]. Additionally, a new informative comment might be already hidden by the user interface thus reducing the chances of users voting on it. As a result, hidden comments may get less public attention. It is unknown how the

[10] https://stackoverflow.com/a/8774101/.
[11] https://meta.stackexchange.com/a/23772.

Table 1. Statistics of Our Studied Data

	Number	Proportion	Mean/Median Score
All Answers	22,668,556	100%	2.6 / 1
$Answer_{comment}$	11,396,766	50.3%	3.9 / 1
$Answer_{noHidden}$	10,093,265	44.5%	3.0 / 1
$Answer_{hidden}$	1,303,501	5.8%	11.1 / 1

comment organization mechanism negatively affects developers in locating relevant information. For example, if a developer is seeking certain information that is actually from a hidden comment, the comment organization mechanism would then lead to information loss. In this sense, it is necessary to examine the actual efficacy of the current comment organization mechanism.

3 DATA COLLECTION

We download the data dump[12] that was published by Stack Exchange in September 2017. Table 1 lists the statistics of our studied data. We focus our study on the answers that have comments (i.e., $Answer_{comment}$). There are 32.2 million comments associated with these $Answer_{comment}$, and 1.3 million (i.e., 11.4%) of $Answer_{comment}$ are answers with hidden comments (i.e., $Answer_{hidden}$). We identify such hidden comments by applying the comment organization mechanism [50] to the collected data in the data dump. We also randomly checked 100 answers on Stack Overflow and all of their associated comments are shown/hidden using the documented mechanism. Under such $Answer_{hidden}$, 4.4 million (i.e., 40.5%) of the comments are hidden. Note that $Answer_{comment}$ includes both $Answer_{hidden}$ and answers that have comments but none of these comments are hidden (i.e., $Answer_{noHidden}$). In general, $Answer_{hidden}$ are more popular than $Answer_{noHidden}$ on Stack Overflow. More specifically, the mean score of $Answer_{hidden}$ is 3.7 times higher than $Answer_{noHidden}$.

4 STUDYING THE INFORMATIVENESS OF COMMENTS (INCLUDING BOTH HIDDEN AND SHOWN COMMENTS)

4.1 RQ1: What are the characteristics of both hidden and shown comments?

Motivation: As shown in Section 2, comments can be informative, thus augmenting their associated answers with valuable knowledge. Identifying what developers discuss in comments helps us better understand how comments actually augment their associated answers, thus providing insights for improving the current commenting system and helping developers retrieve information on Stack Overflow. Stack Overflow uses a comment organization mechanism to split comments under answers into two groups: hidden and shown comments. However, it is not clear whether the shown comments are really more informative than hidden comments as designed. To have a better understanding of hidden and shown comments, we first conduct a quantitative study to investigate the characteristics of the textual content in both hidden and shown comments. Then we conduct a qualitative study to understand whether comments (including both hidden and shown comments) are informative. By knowing this, we can gain insight into the efficacy of the current comment organization mechanism.

Approach: To understand what are the characteristics of both hidden and shown comments, we perform both quantitative and qualitative analysis.

[12] https://archive.org/details/stackexchange.

We study the characteristics of both hidden and shown comments using a quantitative approach that is described in the two following steps. We first compare the length of hidden comments with that of shown comments. The length of a comment is measured by the character number of the comment. We use the length of a comment as a baseline metric to reflect how informative a comment is. In each $Answer_{hidden}$, we calculate the median length (L_{hidden}) of all hidden comments within the answer, and the median length (L_{shown}) of all shown comments within the same answer. The length of a comment is widely used to characterize the quality of the comment in other sites (e.g., MetaFilter and YouTube[13]). As observed in prior research in the natural language processing community, sentences with more words often contain more variant information (i.e., larger entropy) [51]. Prior research also observed that a minimum text length is required to reach a stable text coverage [16]. To compare L_{hidden} and L_{shown}, we define the *median length ratio of comments* in a pairwise manner as $Ratio_L = L_{hidden}/L_{shown}$. A value of $Ratio_L = 1$ means that the median length of all hidden comments under an answer is equal to the median length of all shown comments under the same answer. We also compare L_{hidden} with L_{shown} using the Wilcoxon signed-rank test and the Cliff's delta test [17] to determine if there is any statistically significant difference between L_{hidden} and L_{shown}.

Second, to understand whether hidden comments add diverse information to the associated answers in contrast to their shown comments, we use the vector space model (VSM) to calculate the textual similarity between the hidden comments and the $Answer_{hidden}$ versus the similarity between the shown comments and the $Answer_{hidden}$. In VSM, each document is represented by a vector where the dimension of the vector is equal to the number of unique tokens (i.e., words) in all the documents (i.e., corpus), and the value of each element in a vector is represented by the term frequency–inverse document frequency weight (i.e., TF-IDF). VSM is commonly used for measuring the textual similarity between software engineering artifacts. Readers may refer to the prior studies [14, 20, 41, 54, 63, 65] for more details on VSM.

To apply VSM, we treat each answer as one document, all of its associated hidden comments together as one document, and all of its associated shown comments together as one document. For each document, we first perform the following common pre-processing steps [14, 63]: Remove HTML tags/URL, split words by punctuation marks, split words using camel cases, convert upper case letters to lower case letters, and remove stop words. We use the stop words from the NLTK English stop words collection.[14] Note that we remove URLs for the purpose of building the VSM to compare the textual similarity between comments and their associated answer. In our qualitative analysis, we consider these URLs and find that comments pointing to relevant resources through URLs can be informative (see Section 4.1.2 for details). We then convert each pre-processed document to a vector, in which the weight of each element of the vector is calculated based on term frequency (i.e., the frequency of the term in the document) and inverse document frequency (i.e., the reciprocal of the number of documents containing the term). Finally, we compute the cosine similarity between answers and their associated comments (hidden and shown, respectively).

To compare the textual similarity between hidden and shown comments under the same answer, we calculate the cosine similarity in a pairwise manner. In each $Answer_{hidden}$, we calculate the cosine similarity($S_{Answer\ vs.\ Hidden}$) between the answer and all of its associated hidden comments. In the same $Answer_{hidden}$, we calculate the cosine similarity($S_{Answer\ vs.\ Shown}$) between the answer and all of its associated shown comments. We define the *pairwise cosine similarity ratio* as $Ratio_S = S_{Answer\ vs.\ Shown} \ / \ S_{Answer\ vs.\ Hidden}$. Note that a value of $Ratio_S > 1$ means that the

[13] http://ignorethecode.net/blog/2009/09/29/comments_size_does_matter/.
[14] https://www.nltk.org/book/ch02.html.

shown comments are more similar to their associated answer as compared to that of the hidden comments.

We study the characteristics of both hidden and shown comments using a qualitative approach. To capture the information that developers could obtain from comments, we investigate what is discussed in hidden and shown comments. To do so, we randomly select 384 hidden comments from $Answer_{hidden}$, and randomly select 384 shown comments from $Answer_{hidden}$ to obtain a statistically representative sample with a 95% confidence level and a 5% confidence interval [8].

We manually label the category of each comment. If a comment has multiple sentences, then we assign a label to each sentence individually. Therefore, one comment can be assigned with multiple categories, because a developer may discuss more than one category in a comment. We perform a lightweight open coding-like process that is similar to prior studies [45, 46, 73] to identify the category of topics that are discussed in a comment (i.e., *comment category*). This process involves 3 phases and is performed by the first two authors (i.e., A1–A2) in this article:

- Phase I: A1 identifies a draft list of comment categories based on a sample of 50 comments from hidden comments and another 50 comments from shown comments. Then, A1 and A2 use the draft list to label the comments collaboratively, during which the comment categories are revised and refined. For instance, A1 and A2 discussed their individual list of comment categories and created new labels to resolve any disagreement in the names of labels.
- Phase II: A1 and A2 independently apply the resulting categories from Phase I to label all 768 comments (i.e., 384 hidden comments and 384 shown comments). A1 and A2 take notes regarding the deficiency or ambiguity of the already-identified categories when labeling certain comments. Note that new categories are added during this phase if A1 and A2 observe the need for more categories when the existing labels are unable to represent a new comment. Saturation was constantly monitored during the labeling process and was reached after 200 comments. At the end of this phase, we end up with seven categories of comments (see Table 2). Cohen's kappa [23] is used to measure the inter-rater agreement, and the kappa value is 0.72 (measured at the end of Phase II), implying a high level of agreement.
- Phase III: A1 and A2 discuss the coding results obtained in Phase II to resolve any disagreement until a consensus is reached. No new categories are added during this phase with both A1 and A2 agreeing on all categories. We published our labeled data online.[15]

We also compare if there is a statistically significant difference in the categories of hidden and shown comments using Wilcoxon signed-rank test and Cliff's delta test.

4.1.1 Quantitative Analysis. **There is no statistically significant difference between hidden and shown comments in terms of median length.** We plot the distribution of $Ratio_L$ in $Answer_{hidden}$ (as shown in Figure 3). $Ratio_L$ shows a normal distribution with a mean value of 1. Our statistical test results show that there is no statistically significant difference between hidden and shown comments in terms of length (p-value > 0.05). Moreover, to examine whether the hidden comments are more informative or not, we conduct a qualitative analysis in Section 4.1.2. Our results suggest that hidden comments are as informative as shown comments.

In the majority of $Answer_{hidden}$, hidden comments share less semantic similarity with the associated answers than that of shown comments. In 73.8% of $Answer_{hidden}$, the cosine similarity between the shown comments and the associated answers are no less than the cosine

[15] https://bit.ly/3gobcBO.

Table 2. Comment Categories

Category	Explanation	Example
Praise	Praise an answer	Thank you. It worked for me :)[1]
Advantage	Discuss the advantage of an answer	This is ES6 syntax, it works fine in a sufficiently modern browser.[2]
Improvement	Make improvement to an answer	This should be an OR condition. It can't be both Saturday AND Sunday.[3]
Weakness	Point out the weakness of an answer	@ChrisPatterson I tried that but it would give me errors when I did anything but use the default settings[4]
Inquiry	Make inquiry based on an answer	What's the difference between the two?[5]
Addition	Provide additional information to an answer	an additional point for anyone looking at this. Make sure you haven't set both leading & trailing constraints as those override 'greater than' constraints[6]
Irrelevant	Discuss irrelevant topics to an answer	I'll leave a rebuttal here if that's ok with you. :)[7]

[1]https://www.stackoverflow.com/posts/comments/74500847/.
[2]https://www.stackoverflow.com/posts/comments/63870245/.
[3]https://www.stackoverflow.com/posts/comments/69209752/.
[4]https://www.stackoverflow.com/posts/comments/59923929/.
[5]https://www.stackoverflow.com/posts/comments/66174272/.
[6]https://www.stackoverflow.com/posts/comments/74556909/.
[7]https://www.stackoverflow.com/posts/comments/2771708/.

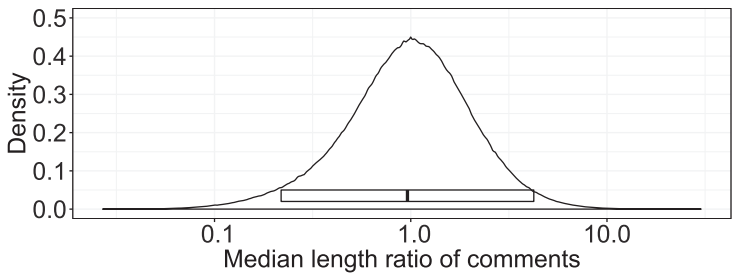

Fig. 3. The distribution of the median length ratio of comments in a pairwise manner ($Ratio_L$). $Ratio_L = 1$ means that the median length of all hidden comments is equal to the median length of all shown comments under the same answer.

similarity between the hidden comments and the associated answers. The Wilcoxon signed-rank test shows a statistically significant difference (p-value < 0.05) for the cosine similarity between shown comments to their associated answers and hidden comments to their associated answers. The Cliff's delta test also shows that the difference is medium (i.e., -0.39).

4.1.2 Qualitative Analysis. **More than half of the comments are informative in both hidden and shown comments.** Except for category *praise* and *irrelevant*, we consider the comments of other categories (i.e., *advantage*, *improvement*, *weakness*, *inquiry*, and *addition*) as *informative*. Note that we consider the category *praise* as redundant for up-voting answers, thus

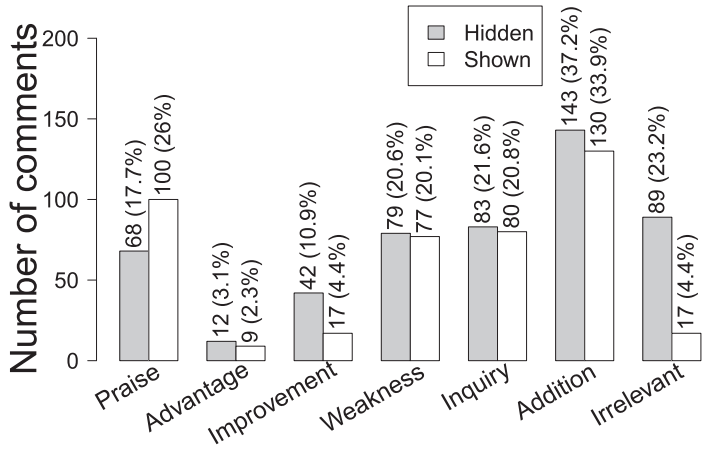

Fig. 4. The distribution of comment categories.

Table 3. Comparison of Hidden and Shown Comments in Providing Additional Value to the Associated Answers

	Informative	Not informative	Total
Hidden comment	280 (72.9%)	104 (27.1%)	384
Shown comment	289 (75.3%)	95 (24.7%)	384

we do not consider such comments as informative. The category *irrelevant* also does not add any direct value to the question answering process.

The distribution of comment categories *advantage*, *weakness*, *inquiry* and *addition* are very similar between hidden and shown comments. The top category is *addition* in both hidden and shown comments. In this category, developers provide additional information to the associated answers of the comment. Namely, by providing an alternative answer to a question, adding an example, adding an explanation, or adding a reference. An example of an informative comment is shown in Figure 1, in which a user pointed out that the node-inspector did not work any more in the latest Node.js version (category *weakness*). He also provides an alternative in the same comment (category *addition*).

There is no statistically significant difference between hidden and shown comments in terms of the distribution of comment categories. Figure 4 shows the distribution of comment categories for both hidden and shown comments. The result of the Wilcoxon signed-rank test shows the differences between hidden and shown comments are insignificant (i.e., p-value $>$ 0.05). The result of Cliff's delta test is negligible (i.e., 0.14). The comparison of the proportion of informative comments in both hidden and shown comments is shown in Table 3. The studied hidden comments share similar information (in terms of the comment categories) compared to shown comments. That being said, *hidden comments are as informative as the shown comments for their associated answer.*

We also note that commenters with higher reputation points are more likely to post more informative comments. In the studied hidden comments, the median reputation points of these commenters are 595.5 and 457.5 for the informative and uninformative comments, respectively. In the studied shown comments, the median reputation points of these commenters are 677 and 364 for the informative and uninformative comments, respectively.

Hidden comments have a similar amount of text as shown comments. Hidden comments share less semantic similarity with the associated answers than that of shown comments in the majority of $Answer_{hidden}$. Hidden comments are as informative as shown comments. More than half of the comments are informative in both hidden and shown comments.

5 STUDYING THE EFFICACY OF THE COMMENT ORGANIZATION MECHANISM

In Section 4, we discover that the majority of the hidden comments are at least as informative as the shown comments. Thus, although the comment organization mechanism has been designed to hide uninformative comments, many informative comments are also hidden.

In addition, developers commonly make use of web search engines, such as Google, to locate online resources to improve their productivity [68]. However, Google does not index such a large number of hidden comments instead it only indexes the shown comments, which prevents developers from accessing the information in these hidden comments from search engines. Therefore, we focus on studying the current comment organization mechanism, that is, the principle that determines whether a comment should be hidden or shown. By investigating the efficacy of the current comment organization mechanism, we wish to offer deeper insights into enhancing the Stack Overflow commenting system, so that developers can more conveniently and effectively perceive informative discussions through comments.

5.1 RQ2: How effective is the current comment organization mechanism?

Motivation: Stack Overflow's current comment organization mechanism aims at showing comments with higher scores while hiding ones with lower scores. The assumption is that comments with higher scores are more informative than the ones with lower scores. However, in Section 4 we find that hidden comments are as informative as shown comments, which suggests that the current comment organization mechanism is not working as expected. Therefore, it is important to investigate the reason behind this.

During our manual study, we also notice that this comment organization mechanism may not work well if comments do not have a hierarchy of different scores. For example, if many comments do not get any up-vote (i.e., their scores are 0), apparently, the current comment organization mechanism would not work in such a scenario. More generally, as long as comments have the same score, the comments would not be ranked nor shown based on their scores (i.e., tie-scored comments). As an example, in an answer,[16] there are 12 comments with only 1 comment with a non-zero score (i.e., 1) as shown in Figure 5. In such cases, the shown comments may not be more informative than the hidden comments.

To study the efficacy of the comment organization mechanism, in this RQ, we investigate how comments are actually ranked and therefore shown.

Approach: Intuitively, an uninformative comment (i.e., category *Praise* and *Irrelevant*) should be hidden by the comment organization mechanism so that another informative comment (such as category *Improvement* and *Weakness*) could be shown under the same answer. The comment organization mechanism is designed for this purpose, i.e., re-arrangement of comments based on their scores. To evaluate the efficacy of the comment organization mechanism in action, we first characterize the comment scores and analyze how they affect the comment organization mechanism, since the current comment organization mechanism is designed based on the comment score.

[16] https://stackoverflow.com/a/45446651.

Unfortunately, it throws the error - Trying to get property of non-object. – fuji Aug 1 '17 at 20:12

What is the result of `dd($ad->categoryItems);` ? – Laerte Aug 1 '17 at 20:16

i.imgur.com/ho8GAET.png – fuji Aug 1 '17 at 20:20

Try to change this in CategoryItems Model: `return $this->belongsTo(Category::class, 'category_id');` . But, anyway, the code seems to be right. – Laerte Aug 1 '17 at 20:26

1 Yeah sure, thank you once again :) – fuji Aug 1 '17 at 20:36

show 7 more comments

Fig. 5. An example of an answer where a large proportion (i.e., 11) of the comments under an answer have 0 score and only 1 comment has a score of 1.

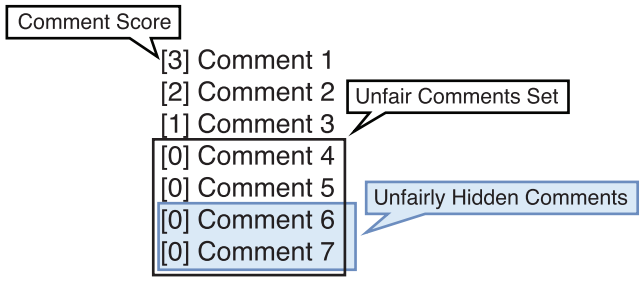

Fig. 6. An example of an unfair comments set and its unfairly hidden comments. Comment 6 and 7 are unfairly hidden comments, since they have the same score as Comment 4 and 5; but were posted at a later date.

More specially, we investigate how the tie-scored comments impact the comment organization mechanism. For this purpose, we make the following definitions. We define that a comment is *unfairly hidden* when it is hidden not because it has a lower score than another comment but because it is posted later than another shown comment with the same score (i.e., *unfairly hidden comments*). In other words, an unfairly hidden comment is hidden because of its later creation time instead of its lower score (note that all such unfairly hidden cases only happen in tie-scored comments based on our definition). We show an example of unfairly hidden comments and unfair comments set in Figure 6. Currently, an unfairly hidden comment occurs in the following situation: for all comments of an answer sorted by score, the score of the sixth comment (i.e., a hidden comment) is equal to the score of the fifth comment (i.e., a shown comment). In this situation, the fifth comment does not need to compete with the sixth comment to be shown by the user interface, it gains its position (as a shown comment) because it is created earlier.

Furthermore, we define a set of comments under an answer as an *unfair comments set*, if there are some hidden and shown comments that have the same score (e.g., a comment with a score of 0 is hidden but another comment with a score of 0 is shown, see Figure 6). We conduct a quantitative study to find out how many answers have unfairly hidden comments. If there were a large proportion of such unfair comments set, then it may indicate that the current comment organization mechanism is not working as expected.

Besides the above-mentioned aspects related to comment score, we calculate the proportion of $Answer_{hidden}$ in which their comments are actually ordered and shown by their creation time (i.e., the comment organization mechanism has no effect). By investigating these characteristics, we wish to understand the impact of comment scores and creation times on the current comment organization mechanism.

Results: Due to the widespread existence of tie-scored comments, unfairly hidden comments exist in 97.3% (i.e., 1,268,416 of 1,303,501) of the $Answer_{hidden}$. Currently, the comment organization mechanism fails to consider tie-scored comments, leading to new comments being hidden while old comments with the same score being shown in almost all $Answer_{hidden}$ (i.e., resulting in the stagnation of showing new comments). Even more, unfairly hidden comments sets have 4,105,956 hidden comments. In other words, 92.9% of all the 4,418,563 hidden comments are actually unfairly hidden (i.e., they are hidden not because of the score or content but the time they are posted). To illustrate the issue, in the same example shown in Figure 5, only 1 comment has a score of 1 while the other 11 comments have a score of 0; therefore, 4 of the 0-score comments are shown simply because they were created earlier than the other 7 comments with 0-score. Therefore, any new comment, even if they are informative, will be automatically hidden. The lack of visibility makes unfairly hidden comments less likely to get any up-voting, and thus are even more likely to remain hidden.

Of $Answer_{hidden}$, 944,950 (i.e., 72.5%) have unfairly hidden comments with a score of 0. More than half (i.e., 56.5%) of $Answer_{hidden}$ have all of their comments with the same score of 0. In other words, as an upper bound estimation, the comment organization mechanism surprisingly only works as expected at most for 43.5% of $Answer_{hidden}$. Moreover, even in such 43.5% cases, it is not guaranteed that every single comment is ranked based on the score, since perhaps only a portion of comments have a score greater than 0. For example, in Figure 5, the answer has 12 comments, and only one of them has a score greater than 0 while the remaining of comments have a score of 0. In this example, the remaining 11 comments are not ranked based on their score anymore. We notice that 87.7% of all the comments under all answers have a score of 0. One possible reason for such a large number of comments that do not have any up-voting is as Calefato et al. [9] mentioned in their previous study, comments are considered as a "free zone" for users, since comments do not generate any reputation point. Thus, users may not be motivated to up-vote comments.

Comments are ranked based on their creation time if their scores are the same. Given the fact that most of $Answer_{hidden}$ have all of their associated comments with the same score, we wish to determine how many $Answer_{hidden}$ have their associated comments actually ordered by the comment creation time.

In 79.4% of $Answer_{hidden}$, comments are ranked and shown by the order of their creation time. In these answers, the result of the comment organization mechanism is equivalent to a queue of comments that are sorted by the creation time of their comments. Namely, only the first 5 oldest comments are shown, and any newer comment will be hidden. In other words, *the current comment organization mechanism gives priority to older comments—promoting stagnation of comments*. As we explained before, one possible reason that a large proportion of $Answer_{hidden}$ are actually shown based on their creation time is the widespread existence of 0-score comments.

Another possible explanation is that older comments tend to get higher scores. Note that the comment age is defined as the time interval between the creation of the comment and its associated answer. Among the 303,035 $Answer_{hidden}$ that have at least 2 comments whose scores are >= 1, 187,714 (i.e., 61.9%) have a negative correlation, and 65,205 (21.5%) have at least a moderate negative correlation (correlation < -0.5) [38] between comment age and score. Among the

46,935 $Answer_{hidden}$ that have at least 5 comments whose scores are >= 1, 36,655 (i.e., 78.1%) have a negative correlation, and 15,525 (i.e., 33.1%) have at least a moderate negative correlation (correlation < -0.5) between comment age and score. Therefore, older comments are more likely to get higher scores.

> The current comment organization mechanism does not work effectively. If an answer has hidden comments, then it is highly likely (97.3%) that it has unfairly hidden comments. The current mechanism fails to consider the widespread of comments with tie-score, especially 0-score, and gives a higher priority to show older comments.

5.2 RQ3: What are the characteristics of unfairly hidden comments?

Motivation: In Section 5.1, by examining the age of comments, their score, and their correlation, we find that, in most cases, the current comment organization mechanism actually fails to rank and show comments based on their scores. The current mechanism does not consider tie-scored comments (i.e., comments that have the same score). For example, during our manual study in Section 4.1.2, we observe that some unfairly shown comments are very short and uninformative (e.g., expressing praise) while some unfairly hidden comments are informative. To improve the current comment organization mechanism, we investigate the characteristics of shown and hidden comments in the unfair comments set. By understanding this, we can provide insightful suggestions for improving the current comment organization mechanism for Stack Overflow.

Approach: We first investigate the length of unfairly shown comments as a baseline metric to measure how informative they are. If some unfairly shown comments are very short, then it is highly likely that they are uninformative. Furthermore, we investigate the informativeness of the shortest unfairly shown comment compared with the longest unfairly hidden comment in the same unfair comments set. The reason that we conduct such comparison is because we probably could provide insights into improving the comment organization mechanism. For example, one simple solution is to replace the shortest unfairly shown comment with the longest unfairly hidden comment if we can show that the longest unfairly hidden comment is more likely to be informative than the shortest unfairly shown comment in the same unfair comments set. To do so, we randomly select 384 sets of comments in the unfair comments sets, with at least 1 unfairly shown comment with length <50 and 1 unfairly hidden comment with length >= 50 to achieve a significance level of 95% and a significance interval of 5%. We use 50 as the text length threshold, because this number well separates between informative and uninformative comments. We use the manually studied comments to analyze the comment length for each comment category. The comment length distribution of comments in each category is shown in Figure 7, where the upper bound of the median comment length for the uninformative comments is 47 (i.e., the irrelevant and praise categories). We manually label the comment category using the same qualitative approach in Section 4, for both the shortest unfairly shown comment and the longest unfairly hidden comment in each sampled set of comments. The Cohen's Kappa value is 0.81 before discussion.

We then perform a qualitative analysis to investigate the comment categories (see Table 2) of each unfairly hidden and unfairly shown comment pair to see if an unfairly hidden comment would be more informative than the corresponding unfairly shown comment under the same answer.

Results: In around half (i.e., 46.6%) of the answers that have unfairly hidden comments, the shortest unfairly shown comments have a length that is less than 50 characters. Figure 8 shows the distribution of answers that have unfairly shown comments against different ranges of the length of the shortest comment under the same answers. Through our observation,

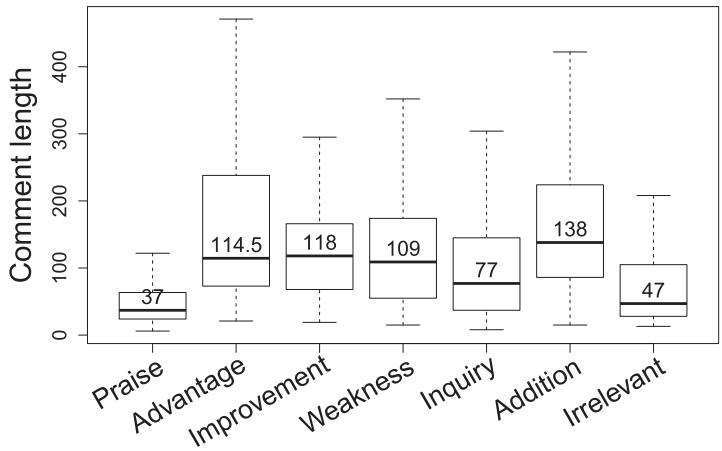

Fig. 7. The distribution of comment length across different comment types.

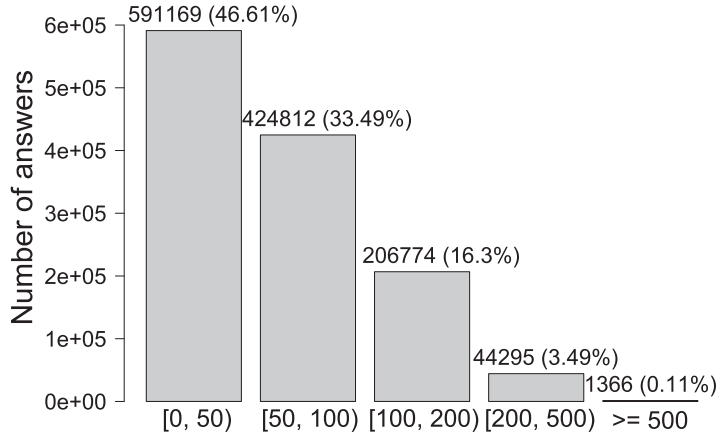

Fig. 8. The distribution of the shortest unfairly shown comments.

we find that short comments are usually not informative. For example, a short comment saying "of course ... that's obvious"[17] does not add any information to the associated answer.

More specifically, in answers that have unfairly hidden comments with the shortest unfairly shown comments being less than 50 characters ($L_{ShownMin}$), we pick the longest unfairly hidden comment ($L_{HiddenMax}$) in the same unfair comments set, and calculate the length ratio as $Ratio_{unfair} = L_{HiddenMax}/L_{ShownMin}$. The distribution of $Ratio_{unfair}$ is shown in Figure 9. *In 63.4% of such cases, the length of the longest unfairly hidden comment is at least 5 times as long as the length of the shortest unfairly shown comment.* Such a high ratio between the longest unfairly hidden comment and the shortest unfairly shown comment of the same answers may indicate that the longest unfairly hidden comment is more informative than the shortest one.

In cases where the shortest unfairly shown comment has fewer than 50 characters in the unfairly hidden comments set, the longest unfairly hidden comment is more likely to be informative than the shortest unfairly shown comment. As shown in Figure 10, in the unfairly hidden comments,

[17] https://stackoverflow.com/posts/comments/56897172.

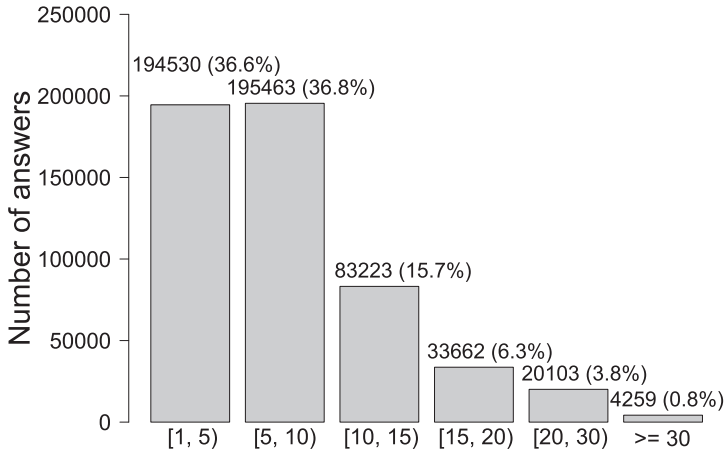

Fig. 9. The distribution of the length ratio.

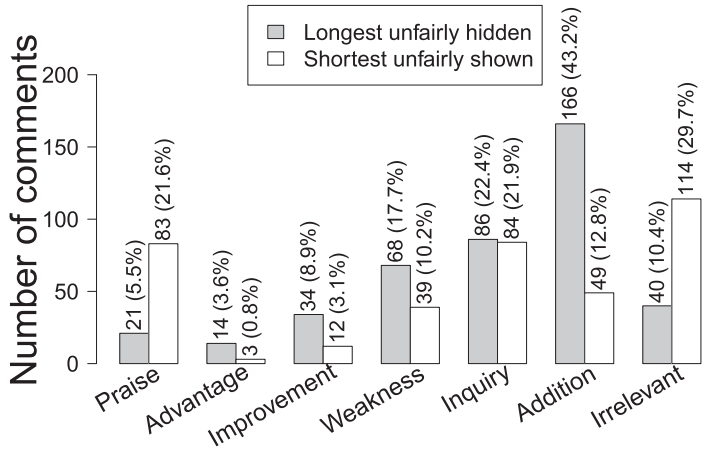

Fig. 10. The distribution of the categories of comments in both the shortest unfairly shown and the longest unfairly hidden comments.

only 15.9% of comments are related to irrelevant information and praise, while in unfairly shown comments 51.3% are related to irrelevant information and praise. As a result, Stack Overflow could replace such short shown comments with another long hidden comment from the unfair comments set.

Discussion: As an exploratory experiment, we inspect how the comment organization mechanism impacts certain informative observations in comments, such as answer obsolescence [73], security vulnerability, and error message. An example of such a comment[18] says, "*works awesome, the only thing that i had to change was the onAttach of the fragment, since it has been deprecated.*" This comment pointed out that the onAttach() function of the Android Fragment class is deprecated; however, this comment is hidden by the comment organization mechanism while none of the currently shown comments bring up this deprecation issue. This motivates us to search among all $Answer_{hidden}$ for comments that mention the word "obsolete" or "outdate," and we find that in

[18] https://stackoverflow.com/posts/comments/78752711.

such 6,523 comments of answer obsolescence observations, 42.5% are actually hidden. *Furthermore, 85.5% (i.e., 2,370) of the 2,771 hidden comments of answer obsolescence observations are actually unfairly hidden by the comment organization mechanism.* Since software obsolescence are more likely to happen over time, the observation of answer obsolescence tends to happen in newer comments instead of older ones. In Section 5.1, we find that in the majority of $Answer_{hidden}$, comments are ranked and shown by their creation time. Therefore, the current comment organization mechanism is much more likely to hide comments that observe answer obsolescence. If these unfairly hidden observations of answer obsolescence could have been replaced by other unfairly shown comments, then developers would be more aware of the answer obsolescence issue on Stack Overflow.

In addition, we find similar trends from other informative observations in comments. For example, among comments mentioning the word "vulnerable" (i.e., 1,603), 38.9% are hidden. Among comments mentioning the word "error" (i.e., 566,756), 36.7% are also hidden.

Among all of the above-mentioned observations that are related to answer obsolescence, security vulnerability, or error message, a significant proportion of such observations are buried in hidden comments. Our finding suggests that the comment organization mechanism does not facilitate developers in finding these obvious flaws in answers. Note that in Section 5.1, we find that the comment organization mechanism only applies to less than half of $Answer_{hidden}$ in the best-case scenario while in the remaining of $Answer_{hidden}$ all comments (due to all of them having a 0-score) are simply ranked and shown by time. Therefore, the comment organization mechanism may bury other informative observations as well.

In the unfair comments set, the longest unfairly hidden comment is more likely to be informative than the shortest unfairly shown comment, especially if the shortest unfairly shown comment has fewer than 50 characters. As a solution to improve the comment organization mechanism, Stack Overflow can swap these pairs of comments.

6 IMPLICATIONS OF OUR FINDINGS

6.1 Implications for Stack Overflow

Crowdsourced knowledge sharing for developers can be undermined due to tie-scored comments in the current comment organization mechanism on Stack Overflow. The comment organization mechanism is unable to prioritize any comment among tie-scored comments. Therefore, these comments could simply be ranked by their creation time, and any new comment other than the oldest 5 can be hidden. Older comments are more likely to be shown and get attention (e.g., have a higher score), while newer comments are more likely to be hidden. The observation exhibits *the Matthew effect: the rich get richer and the poor get poorer* [34]. We observe that 87.7% of all the comments under answers are not upvoted at all (i.e., with a score of 0). One possible explanation is that not everyone can upvote a comment, since at least 15 reputation points are required to upvote a comment. Note that 20.2% of the users have at least 15 reputation points on Stack Overflow. We manually studied the hidden and shown comments and observed that hidden comments are as informative as shown comments, even though such shown comments are ranked higher than hidden comments in terms of their scores. Namely, the score of a comment may not reflect the usefulness of the comment (especially since many informative comments might be already hidden by the user interface in turn reducing the chances of someone voting on them). Hence ranking comments based on their score is likely to lead to biased observations as they are confounded by the fact that a hidden comment is less likely to get voted on.

In the next subsections, we discuss our actionable suggestions in details and the evaluation of them.

6.1.1 How Effective Is It to Classify Informative Comments? One possible solution is to develop an automated classifier to identify informative comments from uninformative comments in the unfair comments set. Although Stack Overflow allows users to up-vote comments, and comments with a higher score are more likely to be among the top 5 thus shown, the current comment organization mechanism does not effectively reinforce its goal. By using this automated approach, the mechanism could be optimized without massive effort of manual labeling by developers. It could also assist developers in flagging comments [73] (i.e., as a way of bringing inappropriate content to the attention of the community,[19] such as labeling unfriendly or unkind comments). Moderators process approximately 1500 flags per day.[20] The classifier could indicate whether a comment is informative from the flagged comments, so he/she can efficiently determine noisy comments for removal and informative comments to keep.

We built a classifier to automatically identify informative comments based on the possible indicators (e.g., the length of a comment) as we observed in RQs. We evaluate our classifier on a dataset of 3,000 manually labeled comments from a prior study [72]. Note that since some comments may have multiple sentences, and some sentences are informative while others are not, we further split a comment into different sentences using periods as separators then label each sentence with the comment category defined in Table 2. In total, we collect 3,654 sentences. As mentioned above, we consider sentences of category *advantage*, *improvement*, *weakness*, *inquiry*, and *addition* as *informative* and sentences of other categories as *uninformative*. We apply the same text preprocessing approach as in RQ1, with a modification to convert any URL address string into a <URL> identifier and to convert any @user_name string into a <CALL> identifier.

Table 4 shows the 13 features that are used in our classifier. Note that we express the textual information of a sentence (i.e., comment text) as an individual feature, since we wish further to examine which feature is important for the built classifier. To capture the textual information of each sentence and express it as an individual feature, we first build a random forest classifier using the unigram TF-IDF score [48] for each word in a sentence as a feature and the label of informative or not for the sentence. Then we collect the prediction score of each sentence that is outputted by the built random forest classifier as the comment text feature to express the textual information for each sentence (for the detail of this approach, see Reference [48]). We use the nltk.sentiment.vader package [40] to calculate the sentiment score for each sentence. Finally, we combine the score of the comment text with the other 12 features to build a second random forest classifier at the sentence level.

We use out-of-sample validation with a bootstrap of 100 iterations to evaluate the effectiveness of our proposed classifier [59, 74]. We measure the effectiveness of our classifier using accuracy, precision, recall, and AUC, which are widely used in the machine learning area for evaluating classifiers [59, 74]. Our classifier achieves 0.86, 0.89, 0.92, and 0.80 in terms of accuracy, precision, recall, and AUC, which demonstrates the effectiveness of our proposed classifier. Note that we consider a comment to be informative if at least one sentence is predicted as informative.

To further understand the important features for our built classifiers, we use the default feature importance calculation technique of random forest classifiers to compute the feature importance. We use the variable importance computation method in the RandomForestClassifier of the scikit-learn package [44]. After computing the variable importance value for each feature across 100 bootstrap iterations, the Scott-Knott clustering [26] is applied to group both the feature importance and the word importance. Comment text has a significant power in identifying the informativeness of a comment. However, the roles of commenters are not important features in our classifier

[19] https://stackoverflow.com/help/privileges/flag-posts.
[20] https://meta.stackexchange.com/a/166628.

Table 4. Features Used in Our Classifier

Dimension	Feature	Type	Explanation
Text	Text score	Numeric	A score to capture textual information
	Length	Numeric	The number of characters in the text
	Sentiment score	Numeric	The sentiment score of the text
Activity	Comment age	Numeric	The time it took to post the comment since the creation of the answer, measured in days
	Comment score	Numeric	The score of the comment
	In an accepted answer	Boolean	Whether the comment is associated with an accepted answer
User	Commenter reputation	Numeric	The reputation points of the commenter before posting the comment
	By asker	Boolean	Whether the comment is posted by the asker of the associated question thread
	By answerer	Boolean	Whether the comment is posted by the answerer of the associated answer
	By another answerer	Boolean	Whether the comment is posted by another answerer of the associated question thread
	By commenter	Boolean	Whether the comment is posted by a commenter who posted an earlier comment in the associated question thread
	By insider	Boolean	Whether the comment is posted by a user who posted the question, any answer, or any comment in the associated question thread before posting the current comment
	By outsider	Boolean	Whether the comment is posted by a user who never posted the question, any answer, or any comment in the associated question thread before posting the current comment

for identifying informative comments compared with other features, suggesting that users post informative comments regardless of their roles in the question thread. Table 5 shows the feature importance of all our features. We also show the top words that contribute to the text score of the comment text in Table 6.

6.1.2 How Effective Is It to Reorganize Comments? One possible approach to alleviate the issue from the comment organization mechanism is to replace shorter shown comments with longer hidden ones. Another possibility is to randomly show comments in the unfair comments set. Other algorithms focusing on sorting tie-scored unfair comments set can be exploited to effectively hide noisy comments while still retaining informative comments.[21] Stack Overflow can also allow for the down-voting of comments to break the tie-scores. This effort can continue to improve the overall quality of knowledge sharing on Stack Overflow.

To explore the effectiveness of these two suggested reorganization mechanisms, we run additional experiments using the cosine similarity to all the 1.3 million $Answer_{hidden}$ with the two reorganization mechanisms:

[21] https://meta.stackexchange.com/q/204402.

Table 5. Feature Importance of the Classifier

SK cluster (5%)	Feature	Means
a	Text score	0.7828
b	Length	0.1037
c	Sentiment score	0.0618
d	Comment age	0.0177
	Commenter reputation	0.0168
e	Comment score	0.0046
	In an accepted answer	0.0038
	By asker	0.0033
	By answerer	0.0015
	By outsider	0.0011
	By insider	0.0011
	By commenter	0.0009
	By another answerer	0.0008

Table 6. Word Importance of the Classifier

SK cluster (5%)	Feature	Means
a	thank	0.0957
b	not	0.0288
	answer	0.0283
	?	0.0271
c	<CALL>	0.0235
	<CODE>	0.0228
d	work	0.0171
e	<URL>	0.0149
f	help	0.0139
	update	0.0131
	good	0.0121
g	use	0.0110
	much	0.0104
	accept	0.0096
h	right	0.0080
	let	0.0077
	but	0.0075
i	edit	0.0071
	great	0.0068
	very	0.0065

- The *Random* mechanism: We replace the unfairly shown comments with randomly selected comments from the unfair comments set (i.e., hidden and shown comments that have the same score).
- The *Length* mechanism: We replace the unfairly shown comments with the longest comments from the unfair comments set.

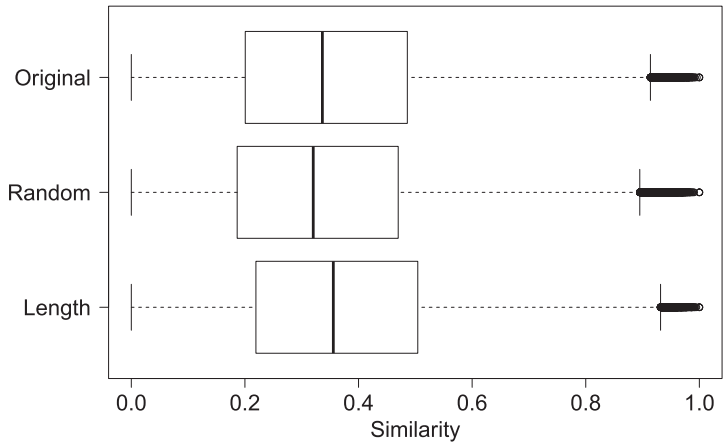

Fig. 11. The distribution of the cosine similarity between answer and shown comments in the different settings.

Figure 11 shows the distribution of cosine similarity between answer and shown comments in the different settings: the original comment organization mechanism, the *Random* mechanism, and the *Length* mechanism. The Wilcoxon signed-rank test shows a statistically significant difference (p-value < 0.05) both for the original comment organization mechanism and the *Random* mechanism, and for the original comment organization mechanism and the *Length* mechanism. For the *Random* mechanism, the similarity between shown comments and answers is decreased, suggesting that users can retrieve more different information from the shown comments compared with the original comment organization mechanism. However, for the *Length* mechanism, the similarity between shown comments and answers increases. Note that shown comments with a longer text length introduce more vocabulary thus it is more likely to overlap with the vocabulary of answers. Therefore, for the purpose of textual diversity, we recommend the *Random* mechanism instead of the *Length* mechanism.

To evaluate our classifier in terms of ranking informative comments, we randomly sample 50 answers with hidden comments, and label the informativeness of their 398 associated comments. We apply our classifier on these labeled comments and calculate the precision/recall at top 5 for these comments. Our classifier achieves a median/mean precision of 100%/89.6% and a median/mean recall of 83.3%/79.3%. In comparison, the original comment organization mechanism has a median/mean precision of 80%/80% and a median/mean recall of 77.5%/70.6%. Thus, our classifier improves the retrieved informative comments in terms of the top 5 comments significantly (p-value < 0.05).

Furthermore, to empirically demonstrate the effectiveness of the two suggested comment reorganization mechanisms, we randomly select 10,000 answers that have unfair comments set, and use the built classifier (in Section 6.1.1) to identify the informative comments from all the associated comments with an answer. Then we calculate the proportion of the shown informative comments over all shown comments of an answer in the different settings of organization mechanisms. If the proportion of the shown informative comments improved in the setting of the suggested mechanisms compared to the current mechanism, then it suggests the effectiveness of our suggested mechanisms. In Figure 12, we observe that in the *Random* mechanism, the proportion of shown informative comments drops compared with the original comment organization mechanism, while in the *Length* mechanism the proportion of shown informative comments increases compared with

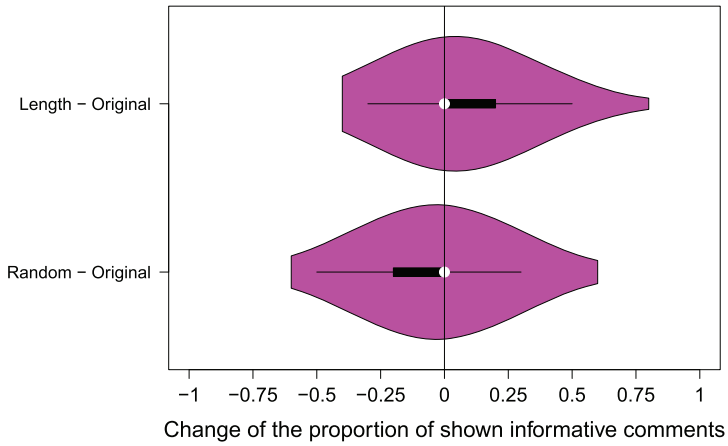

Fig. 12. The distribution of the cosine similarity between answer and shown comments in the different settings.

the original comment organization mechanism. In the *Length* mechanism, 2,994 (i.e., 30.0%) of the answer threads have an increased proportion of shown informative comments, while in the *Random* mechanism, 1,357 (i.e., 13.6%) of the answer threads have an increased proportion of shown informative comments compared with the original comment organization mechanism. We observe that longer sentences tend to contain more explanation/suggestion, while shorter ones are often posted to convey simple thought (e.g., agreement with others). To further evaluate whether the two alternative comment organization mechanisms can improve the proportion of shown informative comments (i.e., the number of shown informative comments out of the shown comments), we randomly select 50 answer threads with hidden comments and manually label whether a comment is informative or not for all the 398 comments that are associated with these answers. We apply both the *Length* and *Random* mechanisms on the labeled comments, and calculate the proportion of shown informative comments for these two alternative mechanisms. We observe that in 12 of the 50 (i.e., 24%) answers, the proportion of shown informative comments is improved by the *Length* mechanism, and in 7 of the 50 (i.e., 14%) answers, the proportion of shown informative comments is improved by the *Random* mechanism. In addition, compared with the original comment organization mechanism, 6,162 (i.e., 61.6%) of the answer threads have a proportion of shown informative comments that remains the same in the *Length* mechanism, while 6,120 (i.e., 61.2%) of the answer threads have a proportion of shown informative comments that remain the same in the *Random* mechanism.

In summary, our explorative analysis shows that the *Random* mechanism is an option to diversify the comments, and the *Length* mechanism enables an improvement to show informative comments.

6.2 Implications for Developers

Developers are encouraged to read through all comments (including hidden comments) in case any further corrections are made in such comments, such as observations of answer obsolescence, security vulnerability, and error messages. The value of Stack Overflow answers can change over time even when these answerers have not yet noticed the change. Therefore, comments provide another channel to notify a wider audience on Stack Overflow about changes to existing answers. Especially in highly attractive answers, many comments are hidden without taking into account

whether they are informative observations to answers. To prevent using an obsolete solution, an insecure code snippet, or a running error, developers are encouraged to read through all comments under an answer before attempting to solve their issues based on the answer, especially more recent ones, because they are more likely to be hidden.

7 THREATS TO VALIDITY

External validity: Threats to external validity are related to the generalization from our study. In this study, we focus on the comment organization mechanism of Stack Overflow, which is the most popular technical Q&A sites in the world. However, our findings and suggestions may not generalize well to other Q&A sites (especially, other sites under Stack Exchange that have the same question, answer, comment layout). Future studies could focus on other Q&A sites, since some of these sites (such as Super User, Server Fault, and Ask Ubuntu) contribute significantly to knowledge sharing in their specific domains.

We conduct two qualitative studies in our case study, the first of which investigates what developers discuss in both hidden and shown comments, and the second one explores whether longer unfairly hidden comments are more informative than shorter unfairly shown comments. Since it is impossible for us to manually study all comments in this study, we attempt to minimize the bias by selecting a statistically representative samples of comments with a 95% confidence level and a 5% confidence interval.

Another threat of our study is that only 5.8% of all answers have hidden comments. Even though the proportion of the $Answer_{hidden}$ is low, we observe that there are *1.3 million* of such answers that have a statistically significantly higher score (i.e., 3.7 times higher, with a mean answer score of 11.1) compared with those without hidden comments (i.e., a mean answer score of 3.0), suggesting that such answers are more recognized by the community. In addition, $Answer_{hidden}$ are associated with question threads that attract 30.2% (i.e., 8.4 billion) of all the viewcounts (i.e., 27.8 billion) from questions across Stack Overflow. Thus, such a small proportion of answers are associated with the question threads that attract a significant amount of traffic on Stack Overflow. Furthermore, these 5.8% answers are associated with question threads that cover 69.2% of all the tags on Stack Overflow.

Internal validity: Threats to internal validity are related to experimental errors. Comment categories are determined by the authors of this study, and later on the informative comment categories are evaluated by the same authors. To reduce the bias of this process, each comment is independently labeled by two of the authors and discrepancies are discussed until reaching a consensus. We provide the level of the inter-rater agreement in our qualitative analysis, and the values of the agreement are high (i.e., 0.72 and 0.81) in both qualitative studies. The authors may not be representative enough to the users on Stack Overflow, and future research is encouraged to conduct user studies to understand what types of comments are perceived to be informative by more users on Stack Overflow.

Construct validity: One threat to construct validity is related to the informativeness of a comment due to the evolution of its associated answer. An informative comment can become uninformative after revision, e.g., the comment is integrated back into the associated answer. However, a prior study shows that answers are not updated frequently after the posting of a comment 85.9% of the answers never get updated after the posting of a comment [72]. Future work is encouraged to further understand how the evolution aspect of comments can affect the informativeness of comments and benefit the question answering processes on Stack Overflow.

In our qualitative study, we identify the informativeness of a comment by taking into account whether the comment can enhance the associated answer. In other words, we identify a comment

as informative if it can complement the original answer. As also noted in a prior work [72], the quality of an answer is a combination of both the answer and its associated comments. Therefore, in this study, we leverage the collection of comments, along with the comment organization mechanism, to gain a deeper understanding of how to enhance the quality of Stack Overflow answers.

Other threats to construct validity are related to our approaches to characterize comments. To compare the degree of information between hidden and shown comments, we first conduct a quantitative analysis. More specifically, we calculate the character length and compute similarity metrics to understand the differences between hidden and shown comments. However, such quantitative measures would not capture all aspects of the differences between hidden and shown comments. Therefore, we also conduct a qualitative study to investigate the difference between hidden and shown comments on top of the quantitative analysis.

In addition, we use a text length threshold of 50 in our study. This value is based on our observation that the upper bound of the median comment length is 47 for uninformative comments. However, we may lose the generalizability of our findings, since a comment with fewer than 50 characters can as well be informative.

8 RELATED WORK

8.1 Improving the Technical Crowdsourced Knowledge Sharing and Management

Stack Overflow is the world's largest crowdsourced knowledge sharing platform. It is essential to deeply understand how developers are sharing their knowledge in this platform. Prior studies investigated how the crowdsourced knowledge is shared on Stack Overflow. For example, Barua et al. [7] used latent Dirichlet allocation to analyze the discussed topics on Stack Overflow, and to discover the trends over time from these knowledge sharing activities. Anderson et al. [3] analyzed the dynamics of knowledge sharing activities on Stack Overflow, such as how answers and votes arrive over time, and whether a question requires a better answer. Wang et al. [62] investigated how users interact with each other on Stack Overflow, and analyzed the behaviors of both answer seekers and answerers to better understand how knowledge is formed for sharing to benefit individual users. Vasilescu et al. [57] studied how users migrate questions from the R user support mailing list (r-help) to crowdsourced knowledge sharing platforms (i.e., Stack Overflow and Cross Validated) and observed that users can get faster answers on crowdsourced sites than on specialized mailing lists. Murgia et al. [39] conducted experiments to simulate a user to answer questions (i.e., an answering bot) on Stack Overflow, and observed that even though the performance of the bot is comparable to human users, humans have a higher expectation from a machine than a human. Upadhyay et al. [56] model the evolution of a user's expertise over time, and observed that prolific learners tend to share knowledge with high value. Mamykina et al. [33] analyzed Stack Overflow's design features and found that its highly responsive and iterative design approach and its incentive system contributed to its success. Baltes et al. [5] investigated how code snippets on Stack Overflow are attributed, and observed that an upper bound of only 25% of code is attributed. While prior research focused on studying the knowledge sharing through the question answering process, our study investigates the knowledge sharing through commenting.

Furthermore, it is essential to understand how the technical crowdsourced knowledge is managed in practice, so that developers can more effectively/efficiently retrieve useful content to solve their daily issues. A number of studies have been done to investigate the quality of questions and answers on Q&A websites [4, 12, 13, 35, 43, 49, 53, 60]. For example, Ponzanelli et al. [43] proposed an approach to identify low quality questions on Stack Overflow. Mizobuchi et al. [35] proposed approaches to improve the detection accuracy of duplicate questions on Stack Overflow. Tausczik et al. [53] investigated how the perceived quality of questions and answers on Math Overflow is

related to the contributor's reputation in the community. Chen et al. [12, 13] studied the collaborative editing activities from Stack Overflow historical post edits, and proposed Neural Network based models to assist post editing activities. Different from prior studies that mainly focus on the quality of questions and answers on Stack Overflow, we focus on studying the quality of comments and identifying informative comments.

A significant number of approaches or tools have been developed to facilitate the knowledge management and retrieving on Q&A websites [15, 21, 25, 29, 32, 37, 64, 69, 71]. For example, to help users identify the most appropriate channel to ask questions, several approaches were developed to help users generate tags automatically when they post questions [64, 69]. Glassman et al. [21] proposed an interactive visualization tool to summarize online code examples at scale. They observed that their interactive visualization helps users answer more API usage questions correctly. Zagalsky et al. [71] developed a code search and recommendation tool for jQuery code search by leveraging the accepted answers related to jQuery on Stack Overflow. Mujumdar et al. [37] proposed an approach to collect and display crowdsourced bug fix suggestions for the Ruby programming language. Ma et al. [32] proposed a multi-layer neural network model to extract API mentions from the informal text in Stack Overflow posts. Hoque et al. [25] developed a tool to analyze textual content, and identify topics and opinions with visualization to support further exploration. Li et al. [29] proposed a recommendation system to provide users with hyperlinks that are highly recognized by the Stack Overflow community. Our study also aims at improving knowledge management and retrieving on Stack Overflow, since informative comments under answers can add value to the knowledge sharing process. More specifically, we characterize both shown and hidden comments, and evaluate the efficacy of the comment organization mechanism. We also provide and evaluate actionable suggestions to help Stack Overflow improve the commenting system toward a better knowledge sharing channel through commenting, e.g., building a classifier to identify informative comments.

8.2 Leveraging Comments on Stack Overflow

Although prior studies of the Stack Overflow ecosystem mainly focus on questions and answers, some studies have taken comments into account. For example, Baltes et al. [6] investigated the evolution of Stack Overflow posts and the temporal relationship between edits and comments. They observed that posts grow over time in terms of the amount of text and the number of code blocks, but the size of a single block is relatively stable. They also observed that some comments may trigger an edit on a post although the correlation is weak (i.e., 0.26). Zhang et al. [72] mined all the comments that are associated with Stack Overflow answers. They analyzed the roles of commenters for different types of comments on Stack Overflow (as shown in Figure 11 of the prior study [72]), and observed that answerers are more likely to post a comment to enhance the associated answers and make clarification. We include the roles of commenters as a feature in our classifier to identify the informative comments. In predicting the long-term value of question threads, Anderson et al. [3] observed that the number of comments in answers has a significant predictive power. Similarly, Tian et al. [55] observed that answers with more comments are more likely to be accepted. Asaduzzaman et al. [4] analyzed both questions and their comments to find out why questions were unanswered. They observed that users may post actual solutions in comments associated with these unanswered questions. Similarly, we observe that users post solutions in comments. Calefato et al. [9] analyzed the sentiment of comments in their study of answer acceptance. They found that the sentiment of comments significantly impacts the chance of answer acceptance. We include the sentiment score of a comment in our classifier and observe that the sentiment score is an important feature after text score and length. Dalip et al. [18] observed that comment can provide additional information to improve the associated posts. In addition, they

found that commenting is useful for measuring the engagement of users in an answer, and this engagement improves the rating of answers. Chang et al. [11] proposed a question routing framework to recommend answerers and commenters to a question. They observed the importance of commenting in further clarifications and the improvement of the quality of an answer. Sengupta et al. [47] examined comments that are associated with 50 posts on Stack Overflow and observed that the discussion is mainly about clarification and modification. Zhang et al. [73] leveraged comments to identify obsolete answers on Stack Overflow, and found that most observations of answer obsolescence in comments are supported with evidence.

The above-mentioned studies extracted heuristic-based features from comments and observed the importance of commenting in the Stack Overflow ecosystem; however, no prior study has specifically investigated and characterized the phenomenon of the comment organization mechanism itself. Furthermore, it is unclear how the mechanism plays a role for developers to retrieve information. Our study examines the informativeness of comments and the effectiveness of the current comment organization mechanism. We wish to offer insights for developers to better use Stack Overflow, and provide actionable suggestions for Stack Overflow engineers to enhance the current commenting system.

8.3 Leveraging User Commentary Content in Software Engineering

To improve the quality of software systems, the software engineering community has proposed many approaches to leverage user's feedback. Since the feedback is based on first-hand experience from end-users, valuable insights can be mined to help developers identify patterns from issues and bugs. Carreño et al. [10] analyzed user comments in the Android Marketplace, and mapped the discussed topics to requirements changes of the mobile applications. Khalid et al. [28] studied user reviews from iOS apps and analyzed how user complaints, such as functional errors and app crashes, negatively affect app ratings in the Apple iOS App Store. Hassan et al. [24] analyzed negative app reviews due to app updates in the Google Play Store, and found that updates with feature removal and user interface issues lead to the highest increase of negative review ratio. Mudambi et al. [36] studied customer reviews on Amazon.com and identified the value of review extremity, review depth, and product type in helping customers make purchasing decisions. Poché et al. [42] found that around 30% of user comments on YouTube videos for coding tutorials are useful to the original videos, and implemented an automated approach to identify useful comments for video creators. Lin et al. [30] studied reviews on the Steam gaming platform, and found that players complain more about game design than software bugs. Vtyurina et al. [58] investigated human behaviour when using chat systems for information seeking. They found that users do not have biases against automatic systems, and users not only look for direct answers but also look for expanded information within a broader context. Wong et al. [66] analyzed the mapping between Stack Overflow posts and code segment, and generated explanatory description automatically for similar code in open-sourced projects.

Similar to previous studies that leverage user feedback in improving existing software artifacts, our study of comments under Stack Overflow answers investigates how both shown and hidden comments add value to their associated answers. Furthermore, we examine whether informative comments are effectively shown using the current comment organization mechanism.

9 CONCLUSION

Stack Overflow uses a comment organization mechanism where at most five comments are shown under each answer. The goal of this mechanism is to improve the compactness of answer threads while retaining the informative comments to facilitate the knowledge sharing process. Among

answers with hidden comments, 40.5% of the comments are hidden by the in-use comment organization mechanism.

In this study, we analyzed 1.3 million answers that have hidden comments to understand the impact of the comment organization mechanism on comments. We found that more than half of hidden and shown comments are informative. In addition, hidden comments are as informative as shown comments, and these hidden comments even add a greater variety of informative content than shown comments to their associated answers.

Furthermore, we evaluated the efficacy of the comment organization mechanism and found that it fails to show informative comments. Comments are unfairly hidden due to the existence of tie-scored comments (especially 0-score comments). Finally, we discuss possible solutions to improve the comment organization mechanism, such as replacing longer unfairly hidden comments with shorter unfairly shown comments. We build a classifier to distinguish informative comments from uninformative comments with an AUC of 0.8. Our analysis shows that the *Random* mechanism can diversify the information in comments, and the *Length* mechanism enables an improvement to show informative comments.

REFERENCES

[1] Rabe Abdalkareem, Emad Shihab, and Juergen Rilling. 2017. On code reuse from Stackoverflow: An exploratory study on android apps. *Inf. Softw. Technol.* 88 (2017), 148–158.

[2] Le An, Ons Mlouki, Foutse Khomh, and Giuliano Antoniol. 2017. Stack overflow: A code laundering platform? In *Proceedings of the 2017 IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER'17)*. IEEE, 283–293.

[3] Ashton Anderson, Daniel Huttenlocher, Jon Kleinberg, and Jure Leskovec. 2012. Discovering value from community activity on focused question answering sites: A case study of Stack Overflow. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'12)*. 850–858.

[4] Muhammad Asaduzzaman, Ahmed Shah Mashiyat, Chanchal K. Roy, and Kevin A. Schneider. 2013. Answering questions about unanswered questions of Stack Overflow. In *Proceedings of the 10th Working Conference on Mining Software Repositories (MSR'13)*. 97–100.

[5] Sebastian Baltes and Stephan Diehl. 2019. Usage and attribution of stack overflow code snippets in GitHub projects. *Emp. Softw. Eng.* 24, 3 (2019), 1259–1295.

[6] Sebastian Baltes, Lorik Dumani, Christoph Treude, and Stephan Diehl. 2018. SOTorrent: Reconstructing and analyzing the evolution of stack overflow posts. In *Proceedings of the 15th International Conference on Mining Software Repositories (MSR'18)*. 319–330.

[7] Anton Barua, Stephen W. Thomas, and Ahmed E. Hassan. 2014. What are developers talking about? An analysis of topics and trends in stack overflow. *Emp. Softw. Eng.* 19, 3 (2014), 619–654.

[8] Sarah Boslaugh. 2012. *Statistics in a Nutshell: A Desktop Quick Reference*. O'Reilly Media.

[9] Fabio Calefato, Filippo Lanubile, Maria Concetta Marasciulo, and Nicole Novielli. 2015. Mining successful answers in Stack Overflow. In *Proceedings of the 12th Working Conference on Mining Software Repositories (MSR'15)*. 430–433.

[10] Laura Carreño and Kristina Winbladh. 2013. Analysis of user comments: An approach for software requirements evolution. In *Proceedings of the 2013 International Conference on Software Engineering (ICSE'13)*. 582–591.

[11] Shuo Chang and Aditya Pal. 2013. Routing questions for collaborative answering in community question answering. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM'13)*. 494–501.

[12] Chunyang Chen, Xi Chen, Jiamou Sun, Zhenchang Xing, and Guoqiang Li. 2018. Data-driven proactive policy assurance of post quality in community q&a sites. In *Proceedings of the ACM on Human-Computer Interaction* (CSCW'18), 1–22.

[13] Chunyang Chen, Zhenchang Xing, and Yang Liu. 2017. By the community & for the community: A deep learning approach to assist collaborative editing in q&a sites. In *Proceedings of the ACM on Human-Computer Interaction (CSCW'17)*, 1–21.

[14] Tse-Hsun Chen, Stephen W. Thomas, and Ahmed E. Hassan. 2016. A survey on the use of topic models when mining software repositories. *Emp. Softw. Eng.* 21, 5 (2016), 1843–1919.

[15] Xiang Chen, Chunyang Chen, Dun Zhang, and Zhenchang Xing. 2019. SEthesaurus: WordNet in software engineering. *IEEE Trans. Softw. Eng.* 1–1. DOI: 10.1109/TSE.2019.2940439

[16] Kiyomi Chujo and Masao Utiyama. 2005. Understanding the role of text length, sample size and vocabulary size in determining text coverage. *Read. Foreign Lang.* 17, 1 (2005), 1–22.

[17] Norman Cliff. 1993. Dominance statistics: Ordinal analyses to answer ordinal questions. *Psychol. Bull.* 114, 3 (1993), 494–509.

[18] Daniel Hasan Dalip, Marcos André Gonçalves, Marco Cristo, and Pavel Calado. 2013. Exploiting user feedback to learn to rank answers in Q&A forums: A case study with Stack Overflow. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'13)*. 543–552.

[19] Rich Gazan. 2010. Microcollaborations in a social Q&A community. *Inf. Process. Manage.* 46, 6 (2010), 693–702.

[20] M. Gethers, R. Oliveto, D. Poshyvanyk, and A. D. Lucia. 2011. On integrating orthogonal information retrieval methods to improve traceability recovery. In *Proceedings of the 2011 27th IEEE International Conference on Software Maintenance*. 133–142.

[21] Elena L. Glassman, Tianyi Zhang, Björn Hartmann, and Miryung Kim. 2018. Visualizing API usage examples at scale. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI'18)*. ACM, New York, NY, Article 580, 12 pages.

[22] Xiaodong Gu, Hongyu Zhang, and Sunghun Kim. 2018. Deep code search. In *Proceedings of the 2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE'18)*. IEEE, 933–944.

[23] Kilem Gwet. 2002. Inter-rater reliability: Dependency on trait prevalence and marginal homogeneity. *Stat. Methods Inter-Rater Reliabil. Assess. Ser.* 2, 1 (2002), 9.

[24] S. Hassan, C. Bezemer, and A. E. Hassan. 2020. Studying bad updates of top free-to-download apps in the Google play store. *IEEE Trans. Softw. Eng.* 46, 7 (2020), 773–793. DOI: 10.1109/TSE.2018.2869395

[25] E. Hoque and G. Carenini. 2014. ConVis: A visual text analytic system for exploring blog conversations. In *Proceedings of the 16th Eurographics Conference on Visualization (EuroVis'14)*. Eurographics Association, Aire-la-Ville, Switzerland, 221–230.

[26] E. G. Jelihovschi, J. C. Faria, and I. B. Allaman. 2014. ScottKnott: A package for performing the Scott-Knott clustering algorithm in R. *J. Land Use Mobil. Environ. (São Carlos)* 15 (04 2014), 3–17. http://www.scielo.br/scielo.php?script=sci_arttext&pid=S2179-84512014000100002&nrm=iso.

[27] Eser Kandogan. 1998. *Hierarchical Multi-window Management with Elastic Layout Dynamics*. Ph.D. Dissertation. University of Michigan.

[28] H. Khalid, E. Shihab, M. Nagappan, and A. E. Hassan. 2015. What do mobile app users complain about? *IEEE Softw.* 32, 3 (May 2015), 70–77.

[29] Jing Li, Zhenchang Xing, Deheng Ye, and Xuejiao Zhao. 2016. From discussion to wisdom: Web resource recommendation for hyperlinks in Stack Overflow. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*. 1127–1133.

[30] Dayi Lin, Cor-Paul Bezemer, Ying Zou, and Ahmed E. Hassan. 2019. An empirical study of game reviews on the Steam platform. *Emp. Softw. Eng.* 24, 1 (2019), 170–207.

[31] Fei Lv, Hongyu Zhang, Jian-guang Lou, Shaowei Wang, Dongmei Zhang, and Jianjun Zhao. 2015. Codehow: Effective code search based on api understanding and extended boolean model (e). In *Proceedings of the 2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE'15)*. IEEE, 260–270.

[32] Suyu Ma, Zhenchang Xing, Chunyang Chen, Cheng Chen, Lizhen Qu, and Guoqiang Li. 2019. Easy-to-deploy API extraction by multi-level feature embedding and transfer learning. *IEEE Trans. Softw. Eng.* 1–1. DOI: 10.1109/TSE.2019.2946830

[33] Lena Mamykina, Bella Manoim, Manas Mittal, George Hripcsak, and Björn Hartmann. 2011. Design lessons from the fastest Q&A site in the west. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'11)*. ACM, New York, NY, 2857–2866.

[34] Robert K. Merton. 1968. The Matthew effect in science. *Science* 159, 3810 (1968), 56–63.

[35] Yuji Mizobuchi and Kuniharu Takayama. 2017. Two improvements to detect duplicates in Stack Overflow. In *Proceedings of the 2017 IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER'17)*. IEEE, 563–564.

[36] Susan M. Mudambi and David Schuff. 2010. What makes a helpful online review? A study of customer reviews on Amazon.com. *MIS Quart.* 34, 1 (2010), 185–200.

[37] Dhawal Mujumdar, Manuel Kallenbach, Brandon Liu, and Björn Hartmann. 2011. Crowdsourcing suggestions to programming problems for dynamic web development languages. In *Proceedings of the CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA'11)*. ACM, New York, NY, 1525–1530.

[38] Mavuto M. Mukaka. 2012. A guide to appropriate use of correlation coefficient in medical research. *Malawi Med. J.* 24, 3 (2012), 69–71.

[39] Alessandro Murgia, Daan Janssens, Serge Demeyer, and Bogdan Vasilescu. 2016. Among the machines: Human-bot interaction on social Q&A websites. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA'16)*. ACM, 1272–1279.

[40] NLTK. 2020. Sentiment Analysis. Retrieved May 2, 2020 from https://www.nltk.org/howto/sentiment.html.
[41] R. Oliveto, M. Gethers, D. Poshyvanyk, and A. De Lucia. 2010. On the equivalence of information retrieval methods for automated traceability link recovery. In *Proceedings of the 2010 IEEE 18th International Conference on Program Comprehension*. 68–71.
[42] Elizabeth Poché, Nishant Jha, Grant Williams, Jazmine Staten, Miles Vesper, and Anas Mahmoud. 2017. Analyzing user comments on YouTube coding tutorial videos. In *Proceedings of the 25th International Conference on Program Comprehension (ICPC'17)*. 196–206.
[43] L. Ponzanelli, A. Mocci, A. Bacchelli, M. Lanza, and D. Fullerton. 2014. Improving low quality Stack Overflow post detection. In *Proceedings of the 2014 IEEE International Conference on Software Maintenance and Evolution*. 541–544.
[44] scikit-learn. 2020. Feature Importance. Retrieved May 2, 2020 from https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html.
[45] Carolyn B. Seaman. 1999. Qualitative methods in empirical studies of software engineering. *IEEE Trans. Softw. Eng.* 25, 4 (1999), 557–572.
[46] Carolyn B. Seaman, Forrest Shull, Myrna Regardie, Denis Elbert, Raimund L. Feldmann, Yuepu Guo, and Sally Godfrey. 2008. Defect categorization: Making use of a decade of widely varying historical data. In *Proceedings of the 2nd ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*. ACM, 149–157.
[47] Subhasree Sengupta and Caroline Haythornthwaite. 2020. Learning with comments: An analysis of comments and community on Stack Overflow. In *Proceedings of the 53rd Hawaii International Conference on System Sciences*.
[48] Emad Shihab, Akinori Ihara, Yasutaka Kamei, Walid M. Ibrahim, Masao Ohira, Bram Adams, Ahmed E. Hassan, and Ken-ichi Matsumoto. 2013. Studying re-opened bugs in open source software. *Emp. Softw. Eng.* 18, 5 (2013), 1005–1042.
[49] I. Srba and M. Bielikova. 2016. Why is Stack Overflow failing? Preserving sustainability in community question answering. *IEEE Softw.* 33, 4 (Jul. 2016), 80–89.
[50] Stack Overflow. 2009. Comments: Top n Shown. Retrieved May 2, 2020 from https://stackoverflow.blog/2009/04/23/comments-top-n-shown/.
[51] Min Tang, Xiaoqiang Luo, and Salim Roukos. 2002. Active learning for statistical natural language parsing. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Philadelphia, PA, 120–127. DOI: https://doi.org/10.3115/1073083.1073105
[52] Chakkrit Tantithamthavorn, Surafel Lemma Abebe, Ahmed E. Hassan, Akinori Ihara, and Kenichi Matsumoto. 2018. The impact of IR-based classifier configuration on the performance and the effort of method-level bug localization. *Inf. Softw. Technol.* 102 (2018), 160–174. DOI: https://doi.org/10.1016/j.infsof.2018.06.001
[53] Yla R. Tausczik and James W. Pennebaker. 2011. Predicting the perceived quality of online mathematics contributions from users' reputations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'11)*. ACM, New York, NY, 1885–1888.
[54] Ferdian Thung, Shaowei Wang, David Lo, and Julia L. Lawall. 2013. Automatic recommendation of API methods from feature requests. In *Proceedings of the 2013 28th IEEE/ACM International Conference on Automated Software Engineering (ASE'13)*. 290–300.
[55] Qiongjie Tian, Peng Zhang, and Baoxin Li. 2013. Towards predicting the best answers in community-based question-answering services. In *Proceedings of the 7th International Conference on Weblogs and Social Media (ICWSM'13)*.
[56] Utkarsh Upadhyay, Isabel Valera, and Manuel Gomez-Rodriguez. 2017. Uncovering the dynamics of crowdlearning and the value of Knowledge. In *Proceedings of the 10th ACM International Conference on Web Search and Data Mining*. 61–70.
[57] Bogdan Vasilescu, Alexander Serebrenik, Prem Devanbu, and Vladimir Filkov. 2014. How social Q&A sites are changing knowledge sharing in open source software communities. In *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW'14)*. 342–354.
[58] Alexandra Vtyurina, Denis Savenkov, Eugene Agichtein, and Charles L. A. Clarke. 2017. Exploring conversational search with humans, assistants, and wizards. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA'17)*. ACM, New York, NY, 2187–2193.
[59] Shaowei Wang, Tse-Hsun Chen, and Ahmed E. Hassan. 2018. Understanding the factors for fast answers in technical Q&A websites. *Emp. Softw. Eng.* 23, 3 (2018), 1552–1593.
[60] S. Wang, T. P. Chen, and A. E. Hassan. 2018. How do users revise answers on technical Q&A websites? A case study on Stack Overflow. *IEEE Trans. Softw. Eng.* 46, 9 (2018), 1024–1038.
[61] Shaowei Wang and David Lo. 2014. Version history, similar report, and structure: Putting them together for improved bug localization. In *Proceedings of the 22nd International Conference on Program Comprehension*. ACM, 53–63.
[62] Shaowei Wang, David Lo, and Lingxiao Jiang. 2013. An empirical study on developer interactions in StackOverflow. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing (SAC'13)*. 1019–1024.
[63] Shaowei Wang, David Lo, and Julia Lawall. 2014. Compositional vector space models for improved bug localization. In *Proceedings of the 30th IEEE International Conference on Software Maintenance and Evolution*. 171–180.

[64] Shaowei Wang, David Lo, Bogdan Vasilescu, and Alexander Serebrenik. 2018. EnTagRec ++: An enhanced tag recommendation system for software information sites. *Emp. Softw. Eng.* 23, 2 (2018), 800–832.

[65] Shaowei Wang, David Lo, Zhenchang Xing, and Lingxiao Jiang. 2011. Concern localization using information retrieval: An empirical study on Linux kernel. In *Proceedings of the 18th Working Conference on Reverse Engineering, (WCRE'11)*. 92–96.

[66] E. Wong, Jinqiu Yang, and Lin Tan. 2013. AutoComment: Mining question and answer sites for automatic comment generation. In *Proceedings of the 2013 28th IEEE/ACM International Conference on Automated Software Engineering (ASE'13)*. 562–567.

[67] Yuhao Wu, Shaowei Wang, Cor-Paul Bezemer, and Katsuro Inoue. 2019. How do developers utilize source code from stack overflow? *Emp. Softw. Eng.* 24, 2 (2019), 637–673.

[68] Xin Xia, Lingfeng Bao, David Lo, Pavneet Singh Kochhar, Ahmed E. Hassan, and Zhenchang Xing. 2017. What do developers search for on the web? *Emp. Softw. Eng.* 22, 6 (2017), 3149–3185.

[69] Xin Xia, David Lo, Xinyu Wang, and Bo Zhou. 2013. Tag recommendation in software information sites. In *Proceedings of the 10th Working Conference on Mining Software Repositories (MSR'13)*. 287–296.

[70] Di Yang, Aftab Hussain, and Cristina Videira Lopes. 2016. From query to usable code: An analysis of stack overflow code snippets. In *Proceedings of the 2016 IEEE/ACM 13th Working Conference on Mining Software Repositories (MSR'16)*. IEEE, 391–401.

[71] Alexey Zagalsky, Ohad Barzilay, and Amiram Yehudai. 2012. Example overflow: Using social media for code recommendation. In *Proceedings of the 3rd International Workshop on Recommendation Systems for Software Engineering (RSSE'12)*. IEEE Press, Los Alamitos, CA, 38–42.

[72] Haoxiang Zhang, Shaowei Wang, Tse-Hsun Chen, and Ahmed E. Hassan. 2019. Reading answers on Stack Overflow: Not enough! *IEEE Trans. Softw. Eng.* 1–1. DOI: 10.1109/TSE.2019.2954319

[73] H. Zhang, S. Wang, T. P. Chen, Y. Zou, and A. E. Hassan. 2019. An empirical study of obsolete answers on Stack Overflow. *IEEE Trans. Softw. Eng.* (2019).

[74] Jiayuan Zhou, Shaowei Wang, Cor-Paul Bezemer, Ying Zou, and Ahmed E. Hassan. 2020. Studying the association between Bountysource bounties and the issue-addressing likelihood of GitHub issue reports. *IEEE Trans. Softw. Eng.* (2020).

Received October 2019; revised October 2020; accepted November 2020