2017 IEEE/ACM 4th International Workshop on CrowdSourcing in Software Engineering (CSI-SE)

Prioritizing User Feedback from Twitter: A Survey Report

Emitza Guzman
University of Zurich
guzman@ifi.uzh.ch

Mohamed Ibrahim
Technische Universität München
m.ibrahim@tum.de

Martin Glinz
University of Zurich
glinz@ifi.uzh.ch

***Abstract*—Twitter messages (tweets) contain important information for software and requirements evolution, such as feature requests, bug reports and feature shortcoming descriptions. For this reason, Twitter is an important source for crowd-based requirements engineering and software evolution. However, a manual analysis of this information is unfeasible due to the large number of tweets, its unstructured nature and varying quality. Therefore, automatic analysis techniques are needed for, e.g., summarizing, classifying and prioritizing tweets. In this work we present a survey with 84 software engineering practitioners and researchers that studies the tweet attributes that are most telling of tweet priority when performing software evolution tasks. We believe that our results can be used to implement mechanisms for prioritizing user feedback with social components. Thus, it can be helpful for enhancing crowd-based requirements engineering and software evolution.**

***Index Terms*—user feedback; crowd-based requirements engineering; crowd-based software evolution.**

I. Introduction

With a daily average of over 500 million micro-messages, commonly referred to as *tweets*, Twitter is one of the most popular social media platforms. Previous research found that Twitter users write tweets about software applications and that such messages contain valuable information that can be used to drive software evolution and elicit new requirements, such as feature requests, bug reports and feature shortcoming descriptions [1]. With the help of tweets, developers, project managers and analysts can obtain feedback from the *crowd*, i.e., current or potential users of the concerned software that are typically distributed among different locations and have a diverse set of needs and expectations. Such information can help software companies to better understand their users and their needs.

Previous research explored user involvement in requirements and software engineering [2] and used the term *crowd-based requirements engineering* [3] for describing the idea of users contributing to different requirements engineering activities. In this respect, Twitter can be seen as a medium for crowd-based requirements engineering in which users submit their feedback – that can later be transformed into requirements. Similarly, user feedback transmitted on Twitter can be used to identify actions to be performed during software evolution, such as bug fixes or feature enhancements.

Nevertheless, the large amount of Twitter data about software applications, its unstructured nature and varying quality calls for the use of automatic processing techniques [1]. Previous work has investigated methods for automatically analyzing unstructured user feedback, available in large numbers, e.g., [4], [5], [6], [7] [8], [9]. One of the studied techniques is feedback prioritization. Existing work has used weighted functions for prioritizing user feedback and has obtained encouraging results [4]. In this previous research the authors assigned the weights based on their own intuition. In our current work, we surveyed 84 software engineering practitioners and researchers about the attributes present in Twitter messages that they consider most important when prioritizing tweets. We believe that our results can be used by industry and academia for finding the weight values in weighted functions used for prioritizing Twitter messages. Furthermore, the described survey could inspire further studies to analyze the attributes of feedback submitted through other channels, such as app stores or specialized user forums.

II. Background

This Section presents the main concepts mentioned in this work:

Tweet: Twitter message are often referred to as tweets. All tweets are limited to 140 characters by the social media platform.

Tweet attribute: Tweets possess different attributes. We distinguish between explicit and implicit tweet attributes. Explicit attributes are readily available through Twitter and its API[1], whereas implicit attributes need additional computation.

The explicit tweet attributes referred in this work are:

- *Retweets*: A retweet is the republishing of a tweet. The retweet number of a particular tweet allows for an assessment of the tweet's reach. A tweet with a high retweet count will reach many people and could imply that a high proportion of users are reporting the same issue.
- *Likes*: Likes are shows of appreciation towards the concerned tweet. The number of likes provides information about the amount of people that find the tweet interesting or are facing the same issue.
- *Followers and friends*: In Twitter, users can follow other users, friends, or have users following them, followers. The number of followers and friends can be impor-

[1] https://dev.twitter.com/overview/api

978-1-5386-4041-8/17 $31.00 © 2017 IEEE
DOI 10.1109/CSI-SE.2017.4

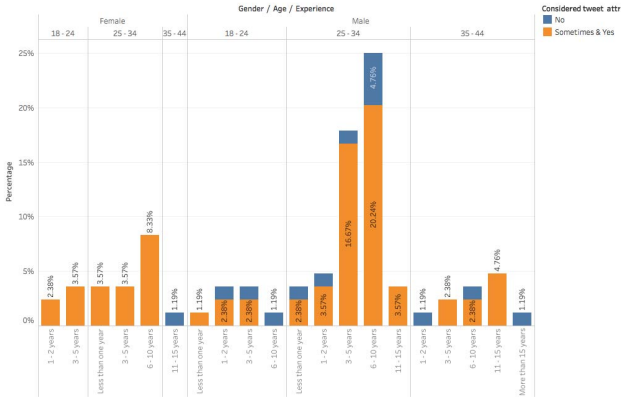

Fig. 1. Breakdown of gender, age, experience and tweet attribute consideration.

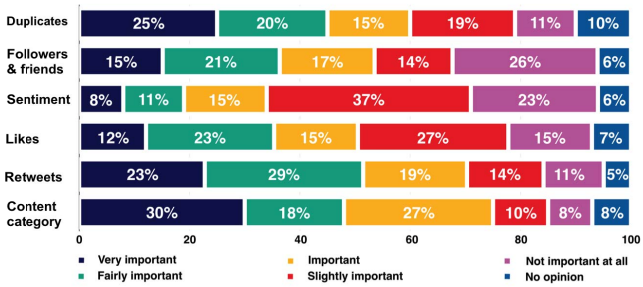

Fig. 2. Tweet attributes that affect tweet prioritization according to surveyed participants.

tant factors when predicting the influence of a Twitter user [10].

The implicit tweet attributes referred in this work are:

- *Duplicates*: the number of tweets that are lexically or semantically similar to a specific tweet. Duplicate tweets could indicate that several users are discussing the same issue.
- *Content category*: the category of the tweet in regards to its content and software evolution. Tweets can contain, for example, "bug reports", "feature requests" or "general praises" [1].
- *Sentiment*: Sentiment is the affect or mood expressed in a tweet. For example, a tweet can have a very positive, neutral or very negative sentiment, or something in between that range.

Prioritizing tweets: In this work, we refer to tweet prioritization as the task of ranking tweets according to their relevance. We define relevance as the factor determining how fast those involved in software evolution (e.g., developers or project managers) should react to the tweet, i.e. a highly relevant tweet should receive immediate attention, whereas the handling of a less relevant tweet can be postponed. A previously used method [4] for prioritizing user feedback are weighted functions. We define the weighted prioritization function, P, prioritizing tweet tw as follows:

$$P(tw) = \sum_{k=1}^{6} w_k * c_k(tw) \quad (1)$$

where c_k are the prioritization coefficients for the six considered attributes of a tweet and w_k are the weights whose value will be studied in the survey described in the following section.

A. Importance of tweet attributes

III. Survey Design

The goal of the survey is to obtain insight about the tweet attributes that affect tweet prioritization according to practitioners and researchers involved in software engineering tasks.

To ensure that all our participants had the necessary background information, we briefly detailed the tweet attributes presented in Section II. Since all tweets used in the survey were derived from Spotify[2], we shortly described the application and its main functionality. Additionally, we asked our participants to imagine themselves in the role of a software developer working at Spotify while answering the survey questions. We chose Spotify as the sample application in the survey, as it is a highly popular application and we believed that most of our survey participants would be familiar with it as end-users.

To give participants a context for the survey, we first asked participants (SQ1) to prioritize ten randomly selected tweets about Spotify. Besides the tweet text, we displayed the tweet attributes (both explicit and implicit, implicit attributes were manually calculated). Second, we asked participants (SQ2) if they had considered the tweet attributes during the prioritization. Third, we requested participants (SQ3) to rate, on a five-level Likert scale (from very important to not important at all), the importance of the displayed tweet attributes when prioritizing the tweets. Moreover, we also asked participants to add any attributes that they considered important for determining tweet relevance that were not included in the list. Next, we asked participants if, having reflected about tweet attributes, they would now prioritize the ten tweets differently (SQ4). A positive answer presented them with the ten tweets they had previously prioritized and asked them to rank them again (SQ5). This question allowed us to examine firsthand whether participants would prioritize tweets differently when their attention is drawn to tweet attributes. It also allowed us to measure the deviation between their old prioritization in which they did not consider tweet attributes and the new one in which they did.

The survey was distributed on the social media of the first two authors. Participants were requested to redistribute the survey, creating a snowball effect.

[2]https://www.spotify.com

TABLE I
ADDITIONAL TWEET ATTRIBUTES FOUND RELEVANT BY SURVEY PARTICIPANTS.

Popularity	Attribute
1.	Payment issue
2.	Actual textual description of reported issue
2.	Revenue of feature being mentioned
2.	Severity of reported issue
3.	Amount of work to fix issue
3.	Combined number of likes and retweets
3.	Gut feeling

IV. SURVEY RESULTS

We report on the 84 complete answers of our survey, all from participants involved in software engineering.

A. Demographics

From the 84 participants, 67% reported being software developers, 15% project managers, 4% product owners, and 15% reported other software engineering roles. The majority of our participants had 6-10 years of experience (38%), followed by those with 3-5 (30%), 1-2 (12%) and 11-15 (10%) years. Moreover, 66% of the participants perform their software engineering tasks in industrial settings, 19% in research institutions, 14% in both research and industry and 1% as a leisure activity. Participants reported 11 places of residence (38% Germany, 14% Switzerland, 13% Egypt, 12% U.S.A, 12% Serbia, 5% Mexico and 6% other countries).

In total, 39% of our participants reported considering tweet attributes when prioritizing the tweets, 44% said that they sometimes considered the attributes and only 17% said that they did not take them into account at all. This result shows that software engineering practitioners and researchers consider tweet attributes significant when prioritizing tweets. Figure 1 shows the breakdown of gender, age and years of experience of participants along with whether they considered tweet attributes when prioritizing the tweets or not.

Figure 2 shows the importance of tweet attributes according to the survey participants. The tweet content category was considered the most important attribute[3] ($\bar{x} = 3.30, s = 1.54$), followed by number of retweets ($\bar{x} = 3.24, s = 1.47$) and number of duplicates ($\bar{x} = 3.01, s = 1.5$). Tweet attributes that were considered less important are social rank ($\bar{x} = 2.68, s = 1.58$), number of likes ($\bar{x} = 2.67, s = 1.47$) and sentiment ($\bar{x} = 2.27, s = 1.32$). Other attributes that users reported as relevant for tweet prioritization are shown in Table I.

The average standard deviation among the answers of all participants in the first prioritization task (SQ1) (μ = 2.42) is slightly larger than when comparing the results of participants of (SQ1) and substituting the rankings of those that reconsidered the role of tweet attributes when prioritizing tweets (SQ5) (μ = 2.37). This difference suggests that the various participants were able to reach a slightly more consensual decision regarding tweet prioritization implying that tweet attributes might eliminate some personal bias.

[3]Reported statistics were performed after transforming the Likert scale categorical values into numerical ones.

V. DISCUSSION

A. Implications

The majority of the participants considered tweet attributes when manually prioritizing tweets. Additionally, participants reported that specific tweet attributes, such as content category, number of retweets and duplicates were more important when prioritizing tweets. This result could derive in distinct weights for some of the specific attributes presented in Section II, when prioritizing informal user feedback with social components by using a weighted function (see Equation 1).

In our future work we will implement a prioritization weighted function using our survey results. Additionally, we will compare the results against weights assigned by our own intuition. Moreover, we will compare the prioritization results obtained by using weighted functions against those obtained when using machine learning techniques.

B. Threats to Validity

The software practitioners and researchers that participated in the survey are not actual developers of the application whose tweets they assessed. Nevertheless, they were familiar with the application as end-users and all reported being involved in software engineering activities. However, their criteria about what constitutes a relevant tweet could be different to that of actual developers of the application.

In our survey we asked participants to explicitly rate the importance of a set of tweet attributes when prioritizing tweets (SQ2). However, there could be a mismatch between what participants report as important with what they actually consider while prioritizing tweets. We addressed this threat by asking participants to prioritize a set of tweets by their relevance (S1) before. We believe that in doing so, participants could become more aware of the tweets attributes on their ranking decision and therefore report on more accurate results.

VI. RELATED WORK

We focus the related work discussion in two areas: the mining of user feedback for software engineering purposes and Twitter in the software engineering domain.

A. Mining User Feedback

User feedback mining has received a considerable amount of attention in recent years. Among the most studied platforms for obtaining user feedback are app stores. Sarro et al. [11] presented a survey of the most relevant work in the area. We focus our discussion on work that could be useful for finding the implicit attributes (*content category*, *duplicates* and *sentiment*) described in Section II. Machine learning approaches have been often applied when automatically categorizing user feedback [12], [13]. Pannichella et al. [14] combined machine learning with linguistic rules for automatically classifying its content, whereas Iacob and

Harrison [8] solely used linguistic rules for classifying feature requests. One of the most common techniques for detecting similar user feedback is topic modeling [5], [6], [8]. On the other hand lexical sentiment analysis has been applied for extracting the affect contained in user feedback [6], [13], whereas machine learning techniques [14] have also been used. We believe that the previously researched tehcniques can be used to extract the implicit attributes mentioned in this work.

Chen et al. [4] and Villarroel et al. [9] studied the prioritization of user feedback. For this purpose, Chen et al. proposed the use of a weighted function and assigned its weights by intuition. Villarroel et al. used machine learning for prioritizing the feedback into different relevance categories. The results from our survey could help assign the weights to functions similar to the ones proposed by Chen et al. Furthermore, additional research is needed to evaluate the performance of weighted functions against traditional machine learning techniques when prioritizing informal user feedabck with social components.

B. Twitter in Software Engineering

Previous work has mainly focused on Twitter use from a technical/development perspective.

With this technical and development focus, previous work has analyzed the automatic processing of tweet information. For example, Prasetyo et al. [15] applied machine learning techniques for distinguishing tweets that mention programming languages as relevant for software development. Achananuparp et al. [16] aggregated tweet content related to programming languages with the help of common topics or keywords. Sharma et al. [17] developed an unsupervised keyword-based approach to identify tweets about software development technicalities.

Singer et al. [18] interviewed and surveyed developers on their Twitter use. They described developers' overload and their difficulties in obtaining relevant information for performing their development tasks. In our previous work [1] we investigated the usage and content characteristics of tweets about software applications and found that the content of some tweets can be relevant for different stakeholders within the project. However, due to their large numbers and varying quality automated analysis processes are needed. In this work we took a step further in this direction and analyzed the importance of specific tweet attributes for prioritizing tweets.

VII. Conclusion

This work reports on a survey with 84 software engineering practitioners and researchers where the importance of specific tweet attributes when prioritizing tweets were analyzed. Our results show that the majority of the participants considered tweet attributes while manually prioritizing tweets and that their consideration can result in slightly larger consensus when manually prioritizing informal user feedback. Additionally, our results show that specific tweet attributes, such as content category, number of retweets and duplicates were considered more important by the participants. Our results can be used for automatically prioritizing informal user feedback with social components. Such mechanisms can be useful for leveraging the crowd to obtain user feedback that can be employed to elicit requirements and drive software evolution.

Acknowledgments We thank Victor Hernández Guzmán and Sofija Hotomski for their help in the survey distribution, as well as our survey participants. This work was partially supported by the European Comission within the SUPERSEDE project (ID 644018).

References

[1] E. Guzman, R. Alkadhi, and N. Seyff, "A Needle in a Haystack: What Do Twitter Users Say about Software?" in *Proc. of the International Requirements Engineering Conference*, 2016, pp. 96–105.

[2] T. Johann and W. Maalej, "Democratic mass participation of users in Requirements Engineering?" in *Proc. of the International Requirements Engineering Conference*, 2015, pp. 256–261.

[3] E. C. Groen, J. Doerr, and S. Adam, "Towards Crowd-Based Requirements Engineering A Research Preview," in *Proc. of Requirements Engineering: Foundation for Software Quality*, 2015, pp. 247–253.

[4] N. Chen, J. Lin, S. C. Hoi, X. Xiao, and B. Zhang, "AR-Miner: mining informative reviews for developers from mobile app marketplace," in *International Conference on Software Engineering*, 2014, pp. 767–778.

[5] L. V. Galvis Carreño and K. Winbladh, "Analysis of user comments: an approach for software requirements evolution," in *Proc. of the International Conference on Software Engineering*, 2013, pp. 582–591.

[6] E. Guzman and W. Maalej, "How do users like this feature? A fine grained sentiment analysis of app reviews," in *Proc. of the International Conference on Requirements Engineering*, 2014, pp. 153–162.

[7] E. Guzman, O. Aly, and B. Bruegge, "Retrieving diverse opinions from app reviews," in *International Symposium on Empirical Software Engineering and Measurement*, 2015, pp. 1–10.

[8] C. Iacob and R. Harrison, "Retrieving and analyzing mobile apps feature requests from online reviews," in *Proc. of the Working Conference on Mining Software Repositories*, 2013, pp. 41–44.

[9] L. Villarroel, G. Bavota, B. Russo, R. Oliveto, and M. Di Penta, "Release planning of mobile apps based on user reviews," in *Proc. of the International Conference on Software Engineering*, 2016, pp. 14–24.

[10] E. Bakshy, J. M. Hofman, W. A. Mason, and D. J. Watts, "Everyone's an influencer: quantifying influence on twitter," in *Proc. of the International Conference on Web Search and Data Mining*, 2011, pp. 65–74.

[11] W. Martin, F. Sarro, Y. Jia, Y. Zhang, and M. Harman, "A Survey of App Store Analysis for Software Engineering," *IEEE Transactions on Software Engineering*, 2016.

[12] E. Guzman, M. El-Halaby, and B. Bruegge, "Ensemble Methods for App Review Classification: An Approach for Software Evolution." in *Proc. of the Automated Software Enginering Conference*, 2015, pp. 771–776.

[13] W. Maalej and H. Nabil, "Bug report, feature request, or simply praise? On automatically classifying app reviews," in *Proc. of the International Requirements Engineering Conference*, 2015, pp. 116–125.

[14] S. Panichella, A. Di Sorbo, E. Guzman, C. Visaggio, G. Canfora, and H. Gall, "How Can I Improve My App? Classifying User Reviews for Software Maintenance and Evolution," in *Proc. of the International Conference on Software Maintenance and Evolution*, 2015, pp. 281 – 290.

[15] P. K. Prasetyo, D. Lo, P. Achananuparp, Y. Tian, and E. P. Lim, "Automatic classification of software related microblogs," in *Proc. of the International Conference on Software Maintenance*, 2012, pp. 596–599.

[16] P. Achananuparp, I. N. Lubis, Y. Tian, D. Lo, and E.-P. Lim, "Observatory of trends in software related microblogs," in *Proc. of the International Conference on Automated Software Engineering*, 2012, pp. 334–337.

[17] A. Sharma, Y. Tian, and D. Lo, "NIRMAL: Automatic identification of software relevant tweets leveraging language model," in *International Conference on Software Analysis, Evolution and Reengineering*. IEEE, 2015, pp. 449–458.

[18] L. Singer, F. Figueira Filho, and M.-A. Storey, "Software Engineering at the Speed of Light: How Developers Stay Current Using Twitter," in *Proc. of the International Conference on Software Engineering*, 2014, pp. 211–221.