2016 IEEE 24th International Requirements Engineering Conference

A Needle in a Haystack: What Do Twitter Users Say about Software?

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

Rana Alkadhi
Technische Universität München & King Saud University
Germany & Saudi Arabia
alkadhi@in.tum.de

Norbert Seyff
FHNW & University of Zurich
Switzerland
norbert.seyff@fhnw.ch

***Abstract*—Users of the Twitter microblogging platform share a vast amount of information about various topics through short messages on a daily basis. Some of these so called tweets include information that is relevant for software companies and could, for example, help requirements engineers to identify user needs. Therefore, tweets have the potential to aid in the continuous evolution of software applications. Despite the existence of such relevant tweets, little is known about their number and content. In this paper we report on the results of an exploratory study in which we analyzed the usage characteristics, content and automatic classification potential of tweets about software applications by using descriptive statistics, content analysis and machine learning techniques. Although the manual search of relevant information within the vast stream of tweets can be compared to looking for a needle in a haystack, our analysis shows that tweets provide a valuable input for software companies. Furthermore, our results demonstrate that machine learning techniques have the capacity to identify and harvest relevant information automatically.**

I. Introduction

Users of the Twitter micro-blogging platform send more than 500 million messages every day[1]. These so-called *tweets* cover a wide range of topics, such as music, television, sports, politics and technology. A quick inspection using the Twitter search functionality shows that users also employ Twitter to communicate about software applications. This could make tweets a relevant source of information for requirements engineers and other stakeholders within software companies. In this respect, tweets could be similar to app reviews, where users recommend software, report on failures and request new features [5], [7], [17]. With the help of tweets, software companies could better understand their users and the users' needs. Furthermore, they could gather information from distributed and remote users, who are typically difficult to involve. Insights gained from tweets could then be used to make informed decisions within software evolution processes.

However, the relevance and impact of tweets for requirements engineering and software evolution, as well as for the different stakeholders within software companies has remained unstudied. A possible reason could be the obfuscation of relevant tweets in the daily flood of messages.

We performed an exploratory study to better understand the communication about software applications on Twitter and its relevance for requirements engineering and software evolution. In this paper, we report on the results of this study in which we collected a dataset of 10,986,494 tweets mentioning 30 popular software applications. We investigated general characteristics of tweets, such as length, frequency and popularity, and used descriptive statistics to report on the results. Furthermore, we randomly selected 1,000 tweets out of the collected dataset and analyzed them manually using content analysis techniques [15]. Finally, we studied the automation potential of the analysis by applying machine learning techniques on the manually analyzed data.

The results of our study demonstrate that tweets contain useful information for software companies. However, due to the large amount of tweets and the high frequency in which they are produced, identifying relevant tweets can be compared to looking for a needle in a haystack. Thus, manually filtering and analyzing relevant tweets is a cumbersome and time-consuming option. Our results also show that automated approaches can filter irrelevant tweets with a precision ranging between 0.78 and 0.54.

The contribution of this work is threefold. First, we detail the usage of Twitter when communicating about software applications. Second, we describe the content present in tweets concerning software applications and its relevance to different stakeholder groups. Finally, we report on an experiment that classifies tweets about software applications according to their relevance for different stakeholders.

II. Study Design

A. Scope and Research Questions

The goal of this study is to explore the status quo of Twitter use when communicating about software applications and to understand the prospect of using tweets for informing requirements engineering and software evolution tasks. For this purpose, we explored the **usage** and **content** of tweets related to software applications, and the **automation potential** of filtering relevant information present in these tweets.

Usage describes how users communicate through Twitter about software applications. In particular, we answered the following research question (RQ):
RQ1 on general characteristics: What are the relevant characteristics of the tweets in terms of frequency, length, popularity

[1] http://www.internetlivestats.com/Twitter-statistics/

2332-6441/16 $31.00 © 2016 IEEE
DOI 10.1109/RE.2016.67

and duplication? Which clients are used for posting tweets and how often do software companies tweet about their software?

Content describes the different semantic categories present in tweets and their characteristics. With this respect, we answered the following questions:
RQ2 on categories: What type of content is present in tweets related to software applications?
RQ3 on relevance: Is the content relevant to software application stakeholders?
RQ4 on sentiment: What are the attitudes that users have when writing about specific content?

Automation potential describes the potential of applying automation techniques in order to process tweet content related to software applications. In this regard, we answered the following question:
RQ5 on classification performance: What is the performance of supervised machine learning techniques when classifying tweets related to software applications according to its relevance for different stakeholders?

B. Dataset

Our dataset consists of 10,986,494 tweets about 30 different desktop and mobile software applications from three different distribution platforms.

We collected tweets for popular mobile applications (apps) available from two of the largest mobile application distribution platforms: Apple's AppStore and Android's Google Play, as well as for desktop applications available in Amazon, a major distributor of desktop applications. We decided to collect data for the ten most downloaded software applications of these platforms (30 in total), as they have a higher probability of being mentioned in a large number of tweets. We leave the study of less popular applications for future work. We obtained the lists of popular applications through charts published by the different distribution platforms[2].

We then used an open-source library[3] to access the Twitter Search API[4] to import tweets written in English which content included the name of at least one of the 30 chosen applications. We imported the tweets for a duration of two months, from November 19, 2015 until January 19, 2016.

Table I shows the selected software applications, their domain and the number of imported tweets for each one. Our dataset includes software products from 14 different domains. Although two of our software products belong to the operating systems domain, we use the term software application to refer to all of them. With the exception of four applications, all have over 1,000 collected tweets.

C. Method

We used descriptive statistics to study the **usage** of Twitter for communicating about software applications. In particular, we analyzed the *general characteristics* of tweets. To study the **content** of tweets, we used content analysis techniques [15] on a stratified random sample of our dataset. Within this analysis, we identified the content *categories* present in tweets and assessed the tweets *relevance* to different stakeholder groups, as well as the *sentiments* of the tweets. We studied the **automation potential** by applying machine learning techniques on the manually analyzed data and measured its *classification performance* according to well established metrics.

TABLE I: Dataset.

Software	Domain	#Tweets
Messenger by Facebook	Social Networking	75,115
Instagram	Photo & Video	1,611,882
Facebook	Social Networking	1,917,568
YouTube	Photo & Video	2,627,979
Snapchat	Photo & Video	2,888,469
Akinator the Genie	Entertainment	1,905
Facetune	Photo & Video	2,644
Architecture of Radio	Education	1,137
Videoshop	Photo & Video	2,249
Afterlight	Photo & Video	8,734
LEO Privacy Guard	Tools	411
Pandora Radio	Music & Audio	59,869
Google Photos	Photo & Video	74,218
Amazon Shopping	Shopping	77,090
Spotify Music	Music & Audio	352,265
Unified Remote Full	Tools	249
Ultimate Guitar Tabs	Music & Audio	705
HotSchedules	Productivity	3,501
WiFi Tether Router	Communication	8
True Skate	Sports	34,765
Kindle	Books	91,683
Norton	Security	156,711
Amazon Music	Music & Audio	135,042
Adobe Photoshop	Photo & Video	32,663
Windows 7	Operating System	158,290
Microsoft Office	Productivity	56,501
McAfee	Security	34,911
Avast	Security	26,376
TurboTax	Finance	14,899
Windows 10	Operating System	538,655
		Total= 10,986,494

III. Usage

In this section, we describe how Twitter users communicate about software applications. We report on the results by applying descriptive statistics.

A. Procedure

When collecting our data, we assembled all tweets which mentioned the name of at least one of the 30 chosen software applications. However, it is possible that tweets stating the software names are unrelated to the software, lack a clear context or contain a large amount of noise. We reduce the probability of reporting on these type of tweets by only considering the software applications where the manual analysis (detailed in Section IV) found that at least 70% of its analyzed tweets were related to the specific software application and had a clear meaning and context[5]. Based on this decision, we

[2] http://www.apple.com/itunes/charts/, https://play.google.com/store/apps/top, http://www.amazon.com/best-sellers-software/zgbs/software

[3] http://www.tweepy.org/

[4] https://dev.Twitter.com/rest/public/search

[5] We consider tweets that do not belong to the *unrelated*, *unclear* or *noise* categories as fulfilling this criteria. A definition of each category can be found in Table II.

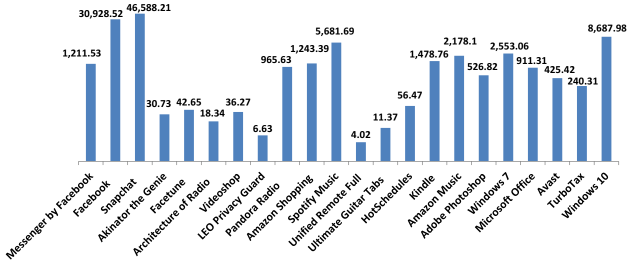

Fig. 1: Daily tweet rate per software application (graph shown in logarithmic scale).

excluded the tweets from the following software applications: Afterlight, Google Photos, Instagram, McAfee, Norton, True Skate and YouTube. Due to the small number of tweets, we also excluded WiFi Tether Router from our analysis. In total, 6,437,286 tweets of 22 software applications were analyzed using descriptive statistics.

B. Results

For most of the software applications considered in this study, a large number of tweets were generated daily. On average, the **frequency** is 31,336.17 tweets per day per software application in our dataset (median=719.06, SD=11,496.78). However, this number varies greatly. While the software application with the highest rate received 46,588.21 tweets per day, the software application with the lowest rate received a considerably sparser amount of 4.02 tweets per day (see Figure 1). Nevertheless, for the majority of the studied software applications, the number of tweets is large enough to make their manual analysis and filtering unfeasible in the long run.

With an average of 13.52 words (median=13, SD=6.39) and 83.41 characters (median=81, SD=36.76)[6], the average **length** of tweets mentioning software applications is comparable to other tweets (average of 15.40 words, 86.30 characters) [10], but shorter than the average review in Apple's distribution platform (106.09 characters) [17]. Although Twitter limits tweet length to 140 characters, users can include photos, videos and links in their tweets to enrich content. In our dataset, 15.1% of the total tweets include links, 4.94% include media, and 4.61% include both. Additionally, Twitter allows for bidirectional communication where users can reply to each other and complement their tweets in case clarifications or further explanations are needed. Reply tweets constitute 22.77% of the tweets in our dataset, indicating a high level of interaction between users speaking about the software applications.

On Twitter, users can react to tweets by *liking* and *re-tweeting*. Liking is used to show appreciation for a tweet, whereas re-tweeting is used to forward tweets to followers. Therefore, re-tweets and likes can be used as indicators of the tweets' **popularity** among the user community. In our dataset, 34.01% of the tweets were liked by other users with an average of 5.06 likes per tweet (median=1, SD=138.34). To a lesser extent, 12.06% of total tweets were re-tweeted by other users with an average of 5.13 re-tweets per tweet (median=1, SD=114.57). Compared to the re-tweeting behavior of random public tweets (2.19%) [25], re-tweeting behavior about software applications is significantly higher.

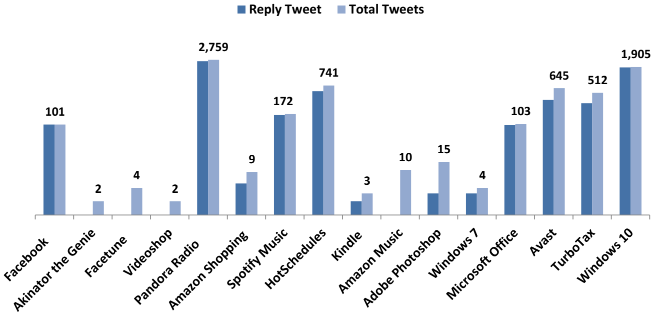

Fig. 2: Total and reply tweets from the software companies to which the analyzed software applications belong (graph shown in logarithmic scale).

Of the 22 analyzed software applications, 21 of their **companies** have an official Twitter account dedicated to the software application[7]. Only four software companies did not post any tweets. Figure 2 shows the number of tweets generated by the software companies and the percentage of reply tweets among them. Less than one percent (0.11%) of the total tweets were tweeted by the software companies. On average, 3.71 tweets are tweeted per day per company (median=0.15, SD=11.51). This result shows that the majority of the studied companies use Twitter to communicate with their users, albeit with different frequency.

Twitter users can post tweets from different software **clients** (e.g., Twitter for iPhone, Instagram, Facebook, Twitterfeed, Twitter Web Client and TweetDeck). Our dataset had an average of 2,994.47 different clients per software application (median=550, SD=1,512.53). A possible interpretation for the large variety of clients is that users tend to post tweets in their current context as soon as they are triggered to post a tweet [21].

To obtain an insight about potentially irrelevant data, we inspected our dataset for **duplicate tweets**, i.e., tweets that have exactly the same text and that are repeated at least once in the dataset. Overall, 9.5% of the tweets in our dataset are duplicate tweets (re-tweets are not included in this count). The average number of duplicate tweets per software application is 192,207.46 tweets (median=2,639.5, SD=71,452.93). One possible cause for tweet duplication is Twitter **bots**, which are automated programs that post tweets with the purpose of spamming or luring users to click on advertisement links. To further explore the link between tweets about software applications and bots, we inspected the users with the maximum number of duplicate tweets per software application. In five of

[6] We follow Twitter's suit and count each link as 23 characters and do not consider media and photos for the count.

[7] Windows 7 and Windows 10 share the common Twitter account @*Windows*.

these cases it were the software companies themselves tweeting about their software. However, the remaining 17 “users” could be identified as bots as their communication patterns followed the Twitter bot communication behavior described by Chu et al. [4]. Furthermore, we found that on average 154.21 different clients per software application include the word ’bot’ as part of their names (median=18, SD=63.66). We hypothesize that the inclusion of ’bot’ in the client’s name could be a reflection of the client’s actual purpose. The proliferation of bots and its association to client names could also explain the high number of clients per software application found in this study.

IV. Content

We manually analyzed tweet content by using the content analysis methods described by Neuendorf [15]. During the analysis, annotators systematically assessed the content of a sample of tweets taken from our dataset, according to an annotation guide. The analysis was conducted by the three authors of the paper. For each analyzed tweet, the three annotators independently assessed the type of *content*, the tweet *relevance* for different stakeholder groups and the *sentiment* of the tweet. In the following sections we detail the analysis procedure and describe the results.

A. Procedure

The content analysis process consisted of five steps:

1) Definition of content categories and stakeholder groups: The aim of this step was to obtain an extensive list of content categories present in tweets and to identify different stakeholder groups who might find distinct tweet content relevant.

For identifying relevant content categories, we used the categories found in a previous study [17] on app reviews as a starting point. This list was extended by adding new categories found by the annotators when individually examining the content of 450 tweets from our dataset (from all software applications). Throughout this process, the annotators provided a definition for each new category, as well as relevant examples. All changes were made available to the other annotators in real-time. Finally, similar categories were merged and their definitions were adapted accordingly. The outcome is a list of 22 categories[8] on what Twitter users say about software (see Table II).

Based on the annotators general knowledge about software engineering and software companies, they identified three different stakeholder groups for whom these categories could be relevant. The groups are defined as follows:

Technical: Stakeholders within the software company who have a strong and direct participation in the software development and evolution process (e.g., requirements engineers, product owners, project managers and developers).

Non-technical: Stakeholders within the software company who have loose participation in the software development and evolution process (e.g., stakeholders from sales, marketing, support, legal and human resource departments).

General public: End-users and potential end-users of the software application.

We discussed the identified stakeholder groups with requirements and software engineering experts. They agreed that our proposal represents a common view on roles and responsibilities within software companies. However, we also concluded that the presented schema might vary from company to company as in addition to the role of stakeholders personal skills and competences are often considered.

2) Annotation guide design: To systematize our manual analysis, we created a guide with definitions and examples of the content categories and sentiment scales, as well as definitions of the different stakeholder groups. To avoid strong disagreements, we conducted three annotation trials of 50 tweets each. After each trial, category and sentiment definitions were slightly refined.

To obtain a shared understanding, annotators discussed the tweet content that could potentially be relevant for each stakeholder group with each other and with a group of requirements and software engineering experts. However, this information was not included in the annotation guide. Therefore, it was up to each annotator to label the relevance of a particular tweet for each stakeholder group. The main reason for this decision was that we wanted to investigate which categories are relevant to the different stakeholder groups based on the actual tweet content and its context - and not on a set of predefined rules.

3) Tweet sampling: We used stratified random sampling to select 33 or 34 tweets per software application, for a total sample of 1,000 tweets[9]. The sampling was applied on our whole dataset (including all 30 software applications). The sample size is similar to other studies performing manual content analysis of software user content [17], [18].

4) Annotation of tweet sample: In this step, the annotators independently labeled each of the 1,000 tweets in the sample. The annotation was done through a specialized web tool that was developed for the task. The tool displayed the name of the software application, name of the user who wrote the tweet and the tweet itself (including clickable links).

Annotators determined the content categories of the tweet, its relevance to the different stakeholders and the sentiment. The tweet sentiments were assessed using a five-level Likert scale ranging from "very positive" (+2) to "very negative" (-2). Annotators labeled the content of each tweet not only based on the content of each tweet, but also considering the content available through links present in the tweet.

Tweets can belong to more than one content category (e.g., a tweet can announce or recommend a software and also mention some of the strengths of its features), hence annotators could label more than one content category for each tweet. Similarly, annotators could label the tweet as being relevant to more than one stakeholder group.

The average time to label the 1,000 tweets was 10.40 hours per annotator. This result corroborates the large amount of

[8] We do not count the categories *unrelated*, *unclear*, *noise* and *other* in this final count.

[9] Tweets samples used in defining content categories and designing the annotation guide were not included in this sample.

TABLE II: Content categories of tweet messages.

Category	Definition
Feature shortcoming	Unsatisfying aspect of an existing feature.
Feature strength	Satisfying aspect of an existing feature.
Feature request	Request for a new feature.
Bug report	Report of an error, flaw, failure or fault.
Usage scenario	A way to use the software (e.g., recommended way, workaround).
Hardware constraint	Hardware needed to run the software.
Software constraint	Software needed to run the software.
General praise	General appreciation of the software focusing on the whole software system.
General complaint	General dissatisfaction of the software focusing on the whole software system.
Advertisement	Promotion of or suggestion to buy the software.
Dissuasion	Advise against the acquisition of the software.
Question	Question directly related to the software.
How to	Explanation to other users how to use the software.
Feature information	Description of a specific feature without any objective evaluation.
Software price	Discussion of the price of the software.
Compliance issue	Dispute over certain terms of agreement or regulations.
Software extension	Description of (planned) extensions of the software.
Other product	Reference to another software product.
Service	Comment on the service provided by the software.
Social interaction	Description of social/personal issues that arise from using the software (i.e., a software feature).
Content related	Comment about content that was created or is available through the software.
Job advertisement	Advertisement of a job available in the company developing the software.
Noise	Tweet not written in English or containing too many illegible symbols to be understandable.
Unclear	Tweet written in English, but the meaning of the tweet is ambiguous or unclear.
Unrelated	Tweet not related to the specific software at all.
Other	Tweet relevant for the study, but not covered by existing categories.

TABLE III: Examples of manual content analysis.

Tweet	Categories	Relevance
I'm glad @HotSchedules is offline but I kind of need to know if my shift got approved or not ????	Bug report	All stakeholders
Facetune – An app to make you good looking.. #Selfies #Photos #Beauty	Advertisement	Non-technical & General public
2000's hip hop radio on pandora	Content related	None
it makes me extremely uncomfortable when people i don't know poke me on facebook	Feature shortcoming & Social interactions	All stakeholders
Surface Pro, which is fine. Just a bit buggy. I'd love a real portable alternative. Wish Adobe would sort out their Photoshop app 2/2	Feature request & Hardware constraint & Other product	All stakeholders

effort required to manually analyze user generated content in the software engineering domain [5], [7], [8].

5) Disagreement handling: As all tweets in the sample were annotated three times, we used a majority voting scheme to resolve relevance and category disagreements. For the tweets where the majority voting results yielded no label (67 tweets), two of the annotators discussed and resolved the disagreements. Sentiment disagreements were resolved by transforming the categorical values into numerical values (in the [-2,2] range) and calculating the median.

B. Results

1) Categories: Table III shows examples of tweets and the categories chosen by the manual annotators for each of them. Each tweet was associated with an average of 1.24 categories (SD=0.46, 217 tweets of the 1,000 sample had more than one category assigned to them). Table IV presents the frequency (in percentage) of each content category. Overall, tweets belonging to the *advertisement* category, which includes the announcement and recommendation of the software, were the most prevalent (28.30%). *Content related* tweets, which mention content managed or produced by the software (25.10%), and tweets that are *unrelated* to the software (15.10%) followed. Categories that are more directly linked to requirements engineering and software evolution tasks were less prevalent, e.g., *bug reports* (0.90%), *feature shortcoming* (1.50%) and *feature request* (0.10%). These percentages seem relatively low. However, if we consider the large amount of tweets that the average application in our data sample receives per day, the numbers are considerable. Assuming that these proportions would hold for a larger sample, the average software application within our dataset would receive, for example, 282 bug reports, 470 feature requests and 31 reports on feature shortcomings on a daily basis. There were no tweets found under the categories *software constraint*, *compliance issue* and *service*

TABLE IV: Manual content analysis results.

		Relevance %			Sentiment	
Category	**Frequency %**	**Technical**	**Non-technical**	**General public**	**Score**	**Interpretation**
Feature shortcoming	1.50	100.00	93.33	93.33	-1.07	negative
Feature strength	0.80	100.00	100.00	100.00	0.03	neutral
Feature request	0.10	100.00	100.00	100.00	0.05	neutral
Bug report	0.90	100.00	88.89	88.89	-0.44	neutral
Usage scenario	2.50	84.00	96.00	84.00	0.00	neutral
Hardware constraint	1.10	27.27	54.55	54.55	0.27	neutral
Software constraint	0.0	N/A	N/A	N/A	N/A	N/A
General praise	2.80	96.43	100.00	100.00	1.21	positive
General complaint	1.10	100.00	100.00	100.00	-1.36	negative
Advertisement	28.30	18.37	98.94	98.94	0.15	neutral
Dissuasion	0.40	100.00	100.00	100.00	-0.25	neutral
Question	0.30	66.67	100.00	100.00	0.00	neutral
How to	3.70	94.59	97.30	97.30	0.00	neutral
Feature information	2.50	76.00	100.00	96.00	0.24	neutral
Software price	8.40	7.14	100.00	100.00	-0.04	neutral
Compliance issue	0.0	N/A	N/A	N/A	N/A	N/A
Software extension	0.10	100.00	100.00	100.00	1.00	positive
Other product	5.90	59.32	88.14	88.14	-0.03	neutral
Service	0.00	N/A	N/A	N/A	N/A	N/A
Social interactions	3.60	25.00	55.56	50.00	0.00	neutral
Content related	25.10	8.37	27.49	37.45	0.09	neutral
Job advertisement	0.30	0.00	100.00	100.00	0.33	neutral
Noise	1.00	0.00	0.00	0.00	0.00	neutral
Unclear	9.70	0.00	0.00	0.00	0.04	neutral
Unrelated	15.10	0.00	0.00	0.00	0.01	neutral
Other	8.10	7.41	49.38	44.44	-0.01	neutral

within our 1,000-sized sample. Therefore, we do not include these categories in the following discussion.

2) Relevance: Tweets were considered relevant to technical stakeholders in 19.30% of the cases, whereas they were assessed as relevant to non-technical stakeholders in 51.50% and to the general public in 53.20% of the cases. Table III shows examples of tweets and the stakeholder groups to which they were considered relevant. Table IV shows the relevance of each category for the specific stakeholder groups. We further analyzed the relevance of each category by analyzing its relevance *tendency* for the different stakeholder groups. We consider that a category has the tendency to be relevant for a specific stakeholder group when more than 80% of the tweets belonging to the category are relevant to the group. Table V shows the categories that tended to be relevant for the different stakeholder groups. Ten categories were relevant for all stakeholder groups. Among these categories are those that are typically linked to software evolution tasks (i.e., *feature shortcoming*, *feature request*, *bug report* and *software extension*), as well as categories that give an idea of user satisfaction (i.e., *general praise*, *general complaint*, *dissuasion* and f*eature strength*) and those that highlight how users use the software (i.e., *usage scenario* and *how to*). Moreover, we also identified six categories which were mainly relevant for non-technical stakeholders and the general public. These categories were mostly related to marketing purposes (i.e., *advertisement*, *other product*, *feature information* and *software price*). Similarly, we identified six categories, which according to our threshold of 80%, cannot be considered relevant for any stakeholder group in general. These categories tended to include either content that was of interest to such a small fraction of people that it was not deemed as interesting for the general public (i.e *social interactions* and *content related*), as well as categories where tweets had no clear meaning (i.e., *unclear* and *noise*) or were unrelated to the software (i.e *unrelated*).

3) Sentiment: Overall, tweets included in the sample tended to be neutral (0.01 average sentiment score). When excluding the tweets that were not related to the software applications (i.e., *noise, unclear, unrelated*) the results remained the same. The large number of tweets with a neutral sentiment could be explained by the proliferation of bot-generated tweets, as described in Section III. As Table IV shows, the tweet categories with the highest positive sentiment polarity were *general praise* (1.21 sentiment score) and *software extension* (1.00 sentiment score). In contrast, the tweet categories with the highest negative polarity were *general complaint* (-1.36 sentiment score), *feature shortcoming* (-1.07 sentiment score) and *bug report* (-0.44 sentiment score). With the exception of the *software extension* category, the results of the tweets with higher sentiment polarity reflect the content nature of the categories: categories highlighting user satisfaction have a positive sentiment, whereas those highlighting user dissatisfaction have a negative sentiment. Analyzing the tweets belonging to *software extension* in more detail we noticed that these tweets were mostly used for marketing purposes by companies that had not written the original software. Contrary to the tweets belonging to the *advertisement* category, which also promoted software or their companies, the tweets belonging to *software extension* seemed to be written by humans.

TABLE V: Relevance tendencies.

Technical, Non-technical and General public		**Non-technical and General public**		**None**	
Feature shortcoming	General praise	Advertisement	Software price	Hardware constraint	Noise
Feature strength	General complaint	Other product		Social interactions	Other
Feature request	Dissuasion	Job advertisement		Content related	
Bug report	How to	Question		Unrelated	
Usage scenario	Software extension	Feature information		Unclear	

V. Automation Potential

The third part of our study consists of an experiment that uses machine learning to classify tweets according to their relevance to the identified stakeholder groups. For training and validating our classifier we used the manually annotated sample described in Section IV. We decided to focus on the classification of tweets according their relevance and not according to their content. This decision was motivated by the high data sparsity in some content categories of our sample, which would make the learning of accurate classifiers for those categories unfeasible.

A. Procedure

A single tweet can be relevant for different stakeholder groups. For example, as Table III shows, the tweet "*it makes me extremely uncomfortable when people i don't know poke me on facebook*" was considered relevant for all different stakeholder groups, whereas the tweet "*Facetune – An app to make you good looking.. #Selfies #Photos #Beauty*" was deemed relevant for the non-technical stakeholders and the general public.

In machine learning, the classification of documents (tweets in our case) into one or more labels (relevance categories in this experiment) is referred to as *multi-label classification*. In our experiment, we used the most popular multi-labeling solution, the *binary relevance method* [28], where a classifier for each label is trained. We compared the performance of two different classifiers: Decision Trees (C4.5 algorithm) and Support Vector Machines (SVM). We chose these classifiers due to their good performance when categorizing text [20].

In order to train the classifiers and report our results, we applied the following steps on our manually annotated tweets:

1) Preprocessing: We preprocessed the tweet text by converting it to tokens and removing stopwords, i.e., common words of the English language that have no specific meaning (e.g., "this", "it", "that"). Additionally, we removed numerical characters and the "#" and "@", common tweet characters, since we considered that they convey little information about the tweet relevance. To further remove unnecessary information, we replaced URLs with a single marker identifying the presence of links in the text.

2) Feature weight conversion: To make tweet text understandable to the different classifiers, we converted the text into a vector space model using TF-IDF [14] as a weighting scheme.

3) Training and evaluation: We applied a 10-fold cross-validation for training the classifiers and evaluating our results. We used three metrics traditionally employed in supervised machine learning for evaluating the accuracy of the classifiers: precision, recall and F-Measure. Their computation is as follows: $Precision_i = \frac{TP_i}{TP_i + FP_i}$ and $Recall_i = \frac{TP_i}{TP_i + FN_i}$. Where TP_i is the total of tweets correctly classified as being relevant to the stakeholder group i, FP_i is the total of tweets incorrectly classified as being relevant to the stakeholder group i and FN_i is the total of tweets that are incorrectly classified as not being relevant to group i. The F-Measure is defined as the harmonic mean of the precision and recall.

B. Results

Table VI gives an overview of the obtained results. Both classifiers had a very similar performance when classifying tweets relevant for non-technical stakeholders and the general public. However, the SVM classifier had a better performance for the classification of the tweets relevant for the technical stakeholders. Overall, the precision and recall values for the prediction of the tweets relevant for the general public and non-technical stakeholders were encouraging (F-measure 0.75). The performance similarity for both stakeholder groups could be due to the fact that, as reported in Section IV-B2, a large number of tweets that are relevant for the non-technical stakeholders are also relevant for the general public. The classification of tweets relevant for technical stakeholders had an F-measure of 0.48 for the SVM classifier, with a precision of 0.54 and a recall of 0.44. Due to the large number of tweets, we argue that it is more important to have higher precision values than those of recall as it will allow more precise filtering of irrelevant information - even at the cost of missing some relevant tweets. Though there is room for improvement, we find this result promising. The classifier is able to accurately filter half of the tweets that are irrelevant for technical stakeholders, which due to the high volume of tweets, could be in the order of thousands per week for popular software applications. We believe that such a classifier could be used as a pre-processing step for a finer-grained classification that, for example, categorizes into the content categories presented in this work or into a subset of them.

One disadvantage of the binary relevance method is the assumption of label independence. Motivated by the apparent inter-relationship between the tweet relevance of non-technical stakeholders and the general public, we compared the binary relevance method against the label powerset method [28], a multi-label classification solution that considers each label combination as a single class. The results, however, were comparable to the ones obtained with the binary relevance, and due to space limitations, are not reported in this work.

TABLE VI: Classification results.

	C4.5 algorithm			SVM		
	Tech.	Non-tech.	Gen. pub.	Tech.	Non-tech.	Gen. pub.
Precision	0.50	0.77	0.78	0.54	0.74	0.74
Recall	0.30	0.73	0.74	0.44	0.77	0.76
F-Measure	0.38	0.75	0.76	0.48	0.75	0.75

VI. Discussion

The results of our study show that (1) tweets contain important information for requirements engineering and software evolution, (2) to use this data for informing requirements engineering and software evolution tasks, automated processing is needed, and (3) automated relevance filtering with a reasonable accuracy is possible. In the following we revisit our research questions.

With respect to the general characteristics of tweets on software applications (*RQ1 on general characteristics*) we conclude that tweets are generated frequently, have a short length and high popularity. Moreover, there are significant duplication levels among the tweets, possibly caused by bots. We also found that most of the studied software companies actively engage in communicating via Twitter about their software applications.

The tweets in our dataset cover several categories (*RQ2 on categories*). Some of these categories have a strong connection to requirements engineering and software evolution tasks, such as *feature shortcomings*, *feature requests*, *bug reports*, *how-tos* and *software extensions*. Nevertheless, their proportion in the whole stream of tweets is relatively low, while still significant due to the large number of tweets. The manual content analysis results show that despite their relative short length, tweets contain relevant information (*RQ3 on relevance*) for requirements engineers and other technical and non-technical stakeholders, as well as for the general public. We found that most users use Twitter to (1) announce or recommend the software applications and (2) post tweets that are related to the content available through the application. Our results show that these tweets are in general not relevant for technical stakeholders. However, in the case of *advertisements* they can be relevant for non-technical stakeholders and for the general public. Overall, the sentiment of tweets (*RQ4 on sentiment*) was neutral, but for tweets expressing satisfaction and dissatisfaction, we could see positive and negative sentiments, respectively.

Although the manual analysis of tweets was a useful technique for our research, automated approaches are needed for analyzing tweets (*RQ5 on classification performance*). This need is motivated by the large number of tweets received daily, the high presence of bot-generated tweets and of tweets that are not relevant for any specific stakeholders (almost 50%). In this respect, our experiment results are encouraging. Tweets can be classified according to their relevance to the different stakeholder groups with a precision ranging from 0.78 to 0.54. In other words, the classifier is able to accurately filter at least half of the tweets that are irrelevant for the concerned stakeholders. Future work could focus on increasing the classifiers' precision by taking additional information about the tweet into consideration (e.g. length, attached media, number of re-tweets, etc). In this work we focused on the classification of tweets based on their relevance and not on the finer-grained categories found in the manual content analysis. We believe that a larger manually annotated set could allow for the training of finer-grained classifiers that could categorize tweets into the content categories described in this work (see Table II). Finally, classifiers trained on the tweets of specific software applications could be evaluated. These classifiers might have a higher performance as they could learn about the specific software context.

Our findings show that Twitter already serves as a communication channel between users and stakeholders within software companies and that the communication of information relevant to requirements engineering and software evolution has started. Twitter has the advantage over other communication channels, such as app stores, that it allows bidirectional communication. This type of communication not only enables users to report issues, but also allows stakeholders within the company to ask questions for clarification and to inform users when issues have been addressed. Experiencing such direct interactions could motivate users to continuously give high quality feedback, thus enabling the evolution of software applications according to user needs. Moreover, the general characteristics of tweets (e.g., frequency, length, number of bot-generated tweets) and the relatively low proportion of tweets that are relevant for technical stakeholders might call for improved or new automatic analysis techniques. Future research needs to investigate to what extent already existing techniques used in other communication channels between stakeholders (e.g., app stores) can be applied to tweets about software applications.

VII. Threats to Validity

Threats to construct validity. For the manual analysis, we rely on error-prone human judgement, as there is a level of subjectivity in deciding if a tweet falls within a specific content category or is relevant to a specific stakeholder group. To address this issue we executed our analysis based on the judgement of three annotators. Furthermore, we created an annotation guide to assure that the annotation task was understood by all annotators, and that all had similar conceptions concerning the content categories, as well as the stakeholder groups. To increase the confidence of the manual analysis results, disagreements were solved by applying a majority voting scheme. Additionally, two annotators discussed and resolved the disagreements of the tweets where the majority voting results yielded no label.

Threats to internal validity. A threat to internal validity involves the list of categories we used for analyzing tweet content. We created our category list by using the content categories found in app reviews from a previous study [17] as a starting point. Then, we modified the list by analyzing the content of 450 tweets and adapting it to include the newly found content. Nevertheless, the list could be incomplete and not reflect the vast amount of information that is mentioned in tweets on software applications. Another threat to validity in our study is the annotation of tweets' relevance, which was determined by the authors of this paper and not by actual stakeholders related to the software companies. Relevance is highly subjective, and even among the actual stakeholders, there could be different understandings of which content is relevant for each of them. We alleviated this threat by discussing possible relevance criteria for each stakeholder group with requirements and software engineering experts before the annotation occurred.

Threats to external validity. We mitigated external validity threats by considering software applications from 14 different domains, for mobile and desktop platforms and from two pricing schemes (paid and free) during the collection of our dataset. Analyzing applications with diversity in these three characteristics allows us to obtain insight about tweet content concerning very different software applications. However, this study did not include applications with low popularity and further research should be conducted to investigate if the results presented in this work hold in that context. Additionally, during our analysis we did not consider special events that could affect the studied tweet characteristics, such as a new release. We relied on manual content analysis to study the tweets' content, its relevance for different stakeholder groups and the automation potential. However, manual analysis on our whole dataset is unfeasible. For this reason, we used a sample of 1,000 tweets. To mitigate generalizability threats, we selected the sample using stratified random sampling, which assured that tweets about the different software applications - with their category and size diversity, were all analyzed in the same degree.

VIII. Related Work

We focus the related work discussion in two areas: Twitter in the software engineering domain and the crowdsourcing of software requirements.

A. Twitter in Software Engineering

Studies of Twitter in the software engineering domain have focused on developers use. There is, to the best of our knowledge, no previous work that has researched its use for explicitly obtaining information to inform requirements elicitation and software evolution processes from an end-user perspective.

Singer et al. [24] surveyed and interviewed developers on their Twitter use. They reported on developers information overload and the difficulty of obtaining relevant content. Our results on the high frequency generation of tweets and the need for automatic processing techniques for finding relevant tweets are inline with their findings.

Previous studies in software engineering have applied content analysis techniques and descriptive statistics to describe tweet content about software development. Bougie et al. [2] manually analyzed tweets posted by software developers and grouped them into different categories. Similarly, Tian et al. [26] manually analyzed the content of tweets mentioning specific programming languages, libraries and systems and methodologies. A follow up study [27] analyzed the frequency, general characteristics and user interaction among Twitter users using the same dataset. Sharma et al. [23] analyzed a set of tweets containing programming language keywords. In their analysis, they automatically detected popular tweet topics and applied content analysis techniques to further investigate popular topics. While the techniques used in this set of previous work are similar to the ones used in our study, the focus is different. We are interested in analyzing information about software applications from a broader perspective that is not necessarily technical or directly related to the software development itself, but that can also include user requirements and experiences with the software - and can therefore help inform requirements engineering and other software evolution tasks.

With this technical and development focus, previous work has also centered on the automatic processing of tweet information. Similar to this study, Prasetyo et al. [19] applied machine learning techniques for identifying tweets that mention programming languages as relevant or irrelevant for software development. Achananuparp et al. [1] aggregated tweet content related to programming languages based on common topics or keywords and built a visualization tool that allows for the analysis of tweet trends. Sharma et al. [22] developed an unsupervised keyword-based approach which detects tweets concerning software development technicalities. We believe that the processing of tweets about software applications could benefit from the aggregation techniques and classification methods detailed in these preceding works.

B. Crowdsourcing Requirements

Previous research [16] found that user feedback is essential for software quality and for identifying ideas of improvement. With the rise of mobile applications and social media, recent research [6], [11] has drawn its attention to the exploration of crowd-based requirements engineering. These works have highlighted the importance of automatic support for processing the elicited feedback. This discussion is inline with the findings of our study, in regard to the need for automatic techniques for the analysis of tweets about software applications.

One of the most studied platforms for obtaining user feedback are mobile distribution platforms. Sarro et al. [13] presented a survey of the most relevant work in the area. In a similar fashion to the study presented in this work, Pagano and Maalej [17] and Hoon [9] conducted exploratory studies and analyzed the amount, content and rating characteristics of user feedback from mobile application distribution platforms. Recent research has also focused on the automatic processing of this feedback. For example, Galvis et al. [5] applied a topic modeling algorithm to automatically extract topics for requirements changes and Chen et al. [3] proposed a framework for mining informative user feedback. Additionally, Guzman and Maalej [8] proposed and approach for extracting features and sentiments mentioned in user feedback from mobile distribution platforms. Previous work [7], [12], [18] used machine learning techniques for the classification of user feedback into categories relevant for software evolution. We believe that the automatic analysis of Twitter messages could benefit from the growing work in this area.

IX. Conclusion

We performed an exploratory study that investigated the use of Twitter while communicating about software applications, the content of tweets about software applications and the automation potential of tweet analysis for requirements engineering and software evolution. We found that tweets contain relevant information for different stakeholder groups. Nevertheless, the proportion of relevant information for technical stakeholders is small compared to the vast amount of received tweets. Thus, automated processes are needed for filtering irrelevant information. The results of an experiment for classifying tweets according to their relevance to different stakeholders show that automated filtering is possible with a reasonable precision ranging from 0.78 to 0.54. Our results demonstrate that Twitter is already being used as a communication channel between users and stakeholders within the software company. We believe that the introduction of further filtering and aggregation mechanisms will allow the incorporation of relevant information regularly submitted by Twitter users into requirements engineering and software evolution processes.

X. Acknowledgements

The authors thank Dustin Wüest, Eya Ben Charrada, Martin Glinz and Melanie Stade for the insightful discussions and valuable feedback. This work was partially supported by the European Comission within the Supersede project (ID 644018).

References

[1] P. Achananuparp, I. N. Lubis, Y. Tian, D. Lo, and E.-P. Lim. Observatory of Trends in Software Related Microblogs. In *Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 334–337, 2012.

[2] G. Bougie, J. Starke, M.-A. Storey, and D. M. German. Towards Understanding Twitter Use in Software Engineering: Preliminary Findings, Ongoing Challenges and Future Questions. In *Proceedings of the 2nd International Workshop on Web 2.0 for Software Engineering (Web2SE)*, pages 31–36, 2011.

[3] N. Chen, J. Lin, S. C. Hoi, X. Xiao, and B. Zhang. AR-Miner: Mining Informative Reviews for Developers from Mobile App Marketplace. In *Proceedings of the 36th International Conference on Software Engineering (ICSE)*, pages 767–778, 2014.

[4] Z. Chu, S. Gianvecchio, H. Wang, and S. Jajodia. Detecting Automation of Twitter Accounts: Are You a Human, Bot, or Cyborg? *IEEE Transactions on Dependable and Secure Computing*, 9(6):811–824, 2012.

[5] L. V. Galvis Carreño and K. Winbladh. Analysis of User Comments: An Approach for Software Requirements Evolution. In *Proceedings of the 2013 International Conference on Software Engineering (ICSE)*, pages 582–591, 2013.

[6] E. C. Groen, J. Doerr, and S. Adam. Towards Crowd-Based Requirements Engineering A Research Preview. In *Requirements Engineering: Foundation for Software Quality*, volume 9013, pages 247–253. 2015.

[7] E. Guzman, M. El-Halaby, and B. Bruegge. Ensemble Methods for App Review Classification: An Approach for Software Evolution. In *Proceedings of the 30th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 771–776, 2015.

[8] E. Guzman and W. Maalej. How do users like this feature? a fine grained sentiment analysis of app reviews. In *Proceedings of the IEEE 22nd International Requirements Engineering Conference (RE)*, pages 153–162, 2014.

[9] L. Hoon, R. Vasa, J.-G. Schneider, J. Grundy, and Others. An Analysis of the Mobile App Review Landscape: Trends and Implications. *Faculty of Information and Communication Technologies, Swinburne University of Technology, Tech. Rep*, 2013.

[10] B. J. Jansen, M. Zhang, K. Sobel, and A. Chowdury. Twitter Power: Tweets as Electronic Word of Mouth. *Journal of the American Society for Information Science and Technology*, 60(11):2169–2188, 2009.

[11] T. Johann and W. Maalej. Democratic Mass Participation of Users in Requirements Engineering? In *Proceedings of the IEEE 23rd International Requirements Engineering Conference (RE)*, pages 256–261, 2015.

[12] W. Maalej and H. Nabil. Bug Report, Feature Request, or Simply Praise? On Automatically Classifying App Reviews. In *Proceedings of the IEEE 23rd International Requirements Engineering Conference (RE)*, pages 116–125, 2015.

[13] W. Martin, F. Sarro, Y. Jia, Y. Zhang, and M. Harman. A survey of app store analysis for software engineering. *RN*, 16:02, 2016.

[14] T. M. Mitchell. *Machine Learning*, volume 4 of *McGraw-Hill Series in Computer Science*. 1997.

[15] K. Neuendorf. *The Content Analysis Guidebook*. Thousand Oaks, CA: Sage Publications, 2002.

[16] D. Pagano and B. Bruegge. User Involvement in Software Evolution Practice : A Case Study. In *Proceedings of the 2013 International Conference on Software Engineering (ICSE)*, pages 953–962, 2013.

[17] D. Pagano and W. Maalej. User Feedback in the Appstore: an Empirical Study. In *Proceedings of the 21st IEEE International Requirements Engineering Conference (RE)*, pages 125–134, 2013.

[18] 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 *Proceedings of the 31st International Conference on Software Maintenance and Evolution (ICSME)*, pages 281–290, 2015.

[19] P. K. Prasetyo, D. Lo, P. Achananuparp, Y. Tian, and E. P. Lim. Automatic Classification of Software Related Microblogs. In *Proceedings of the 28th IEEE International Conference on Software Maintenance (ICSM)*, pages 596–599, 2012.

[20] F. Sebastiani. Machine Learning in Automated Text Categorization. *ACM computing surveys (CSUR)*, 34(1):1–47, 2002.

[21] N. Seyff, G. Ollmann, and M. Bortenschlager. AppEcho: A User-driven, in Situ Feedback Approach for Mobile Platforms and Applications. In *Proceedings of the 1st International Conference on Mobile Software Engineering and Systems (MOBILESoft)*, pages 99–108, 2014.

[22] A. Sharma, Y. Tian, and D. Lo. NIRMAL: Automatic Identification of Software Relevant Tweets Leveraging Language Model. In *Proceedings of the IEEE 22nd International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 449–458, 2015.

[23] A. Sharma, Y. Tian, and D. Lo. What's Hot in Software Engineering Twitter Space? In *Proceedings of the IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pages 541–545, 2015.

[24] L. Singer, F. Figueira Filho, and M.-A. Storey. Software Engineering at the Speed of Light: How Developers Stay Current Using Twitter. In *Proceedings of the 36th International Conference on Software Engineering (ICSE)*, pages 211–221, 2014.

[25] B. Suh, L. Hong, P. Pirolli, and E. H. Chi. Want to be Retweeted? Large Scale Analytics on Factors Impacting Retweet in Twitter Network. In *Proceedings of the IEEE Second International Conference on Social Computing (SocialCom)*, pages 177–184, 2010.

[26] Y. Tian, P. Achananuparp, I. N. Lubis, D. Lo, and E.-P. Lim. What Does Software Engineering Community Microblog About? In *Proceedings of the 9th IEEE Working Conference on Mining Software Repositories (MSR)*, pages 247–250, 2012.

[27] Y. Tian and D. Lo. An Exploratory Study on Software Microblogger Behaviors. In *Proceedings of the IEEE 4th Workshop on Mining Unstructured Data (MUD)*, pages 1–5, 2014.

[28] G. Tsoumakas and I. Katakis. Multi-label Classification: An Overview. *International Journal of Data Warehousing and Mining*, 3:1–13, 2007.