# Can a Conversation Paint a Picture? Mining Requirements in Software Forums

James Tizard, Hechen Wang, Lydia Yohannes, Kelly Blincoe
University of Auckland, New Zealand
{jtiz003,hwan531}@aucklanduni.ac.nz, l.yohannes@web.de, k.blincoe@auckland.ac.nz

***Abstract*—The modern software landscape is highly competitive. Software companies need to quickly fix reported bugs and release requested new features, or they risk negative reviews and reduced market share. The amount of online user feedback prevents manual analysis. Past research has investigated automated requirement mining techniques on online platforms like App Stores and Twitter, but online product forums have not been studied.**

**In this paper, we show that online product forums are a rich source of user feedback that may be used to elicit product requirements. The information contained in forum questions is different from what has been described in the related work on App Stores or Twitter. Users often provide detailed context to specific problems they encounter with a software product and other users respond with workarounds or to confirm the problem. Through the analysis of two large forums, we identify 18 different types of information (classifications) contained in forums that can be relevant to maintenance and evolution tasks. We show that a state-of-the-art App Store tool is unable to accurately classify forum data, which may be due to the differences in content. Thus, specific techniques are likely needed to mine requirements from product forums. In an exploratory study, we developed classifiers with forum specific features. Promising results are achieved for all classifiers with f-measure scores ranging from 70.3% to 89.8%.**

## I. Introduction

In today's modern software landscape, there is a large amount of online reviews and discussions about software products. These reviews and discussions can contain valuable information for the product development team. Manual analysis to extract new requirements from this text is time intensive due to the volume of feedback and challenging due to the varied backgrounds of highly distributed user bases [1]. Existing research has investigated how product development insights can be automatically extracted from platforms like Twitter and App Stores [2], [3], [4], [5], [6], [7], but discussion venues like product forums have not been extensively studied.

Product forums help to facilitate an online community, where users can come for product support or to assist other users. Forums are widely used by large software vendors, with 94.7% of vendors using them to support the use of their applications [8]. While their primary function is to help customers use the application, forums are also a rich source of untapped user feedback. Forum posts often contain negative user experiences, issues with the product, descriptions of new desirable features, or ways the product is unintuitive to use.

In this paper, we describe how product forums differ from app reviews and tweets in both their content and structure. Forums are structured as a back and forth discussion between the product users and owners, whereas app reviews are more unidirectional. Also, forum posts often contain more text and therefore contain more information than the short explicit app reviews or tweets. We analysed forums of two popular software products: the VLC media player and the Firefox browser. Both have active user support communities and a large number of forum posts, making them an ideal research setting.

The contributions of this paper are threefold: (1) We describe the information contained within software product forums. A formal categorisation of forum sentences that relate to software evolution and maintenance is presented. These classifications are more fine-grained than categorisations developed in previous research on App Store reviews and Twitter and include several new forum specific classes. (2) We show that additional requirement mining techniques are required for software product forums. We evaluate a state-of-the-art App Store tool, ARdoc [2], on its ability to classify forum sentences. ARdoc was chosen due to it's excellent performance in app review classification and its user friendly interface. However, we found it is insufficient when applied to forums. One possible reason for this is the differences in content and structure across the two sources. (3) We describe new forum specific features and the application of novel machine learning techniques, developed through an exploratory study of the VLC and Firefox forums. We show that classifiers built specifically for forum classification using these features and techniques outperform the existing App Store tool when applied to forums.

Our analysis was guided by the following research questions:

**RQ1:** *What user feedback is contained in the VLC and Firefox product forums?*

**RQ2:** *Can a tool developed to classify and extract information from App Store reviews be used to classify forum sentences?*

**RQ3:** *Can a classifier that utilises characteristics specific to user forums outperform a classifier developed for App Store reviews?*

The paper is structured as follows: section II reviews the related work that informed our research. Section III describes our research setting. In sections IV, V and VI we present the research methodology and results for each research question, respectively. Our findings are discussed in section VII. Section VIII discusses the threats to the validity of our findings and, finally, section IX concludes the paper.

## II. Background

### A. Motivation to mine online user feedback

There has been significant research on the usefulness of analysing user feedback. User feedback can be critical to a product's success. Harman et al. [9] showed the importance of positive user feedback, finding a strong correlation between customer ratings and the popularity of a mobile app.

Feedback can also be important for product development insights. Pagano and Bruegge [10] surveyed developers and found that user feedback contains important information for developers, helps to improve software quality and to identify missing features. Pagano and Maalej [11] found that approximately a third of user reviews in App Stores contain information related to software requirements, including bug reports, feature requests and user experiences.

Yet, assessing the relevance of and extracting the important feedback is mostly accomplished manually, which is very resource intensive [10]. Many researchers describe the challenges of mining product requirements from a large, highly distributed user base and emphasise the need for tool-based assistance (e.g., [1], [10]).

### B. Existing requirement mining from user feedback

Much research has investigated how requirement information can be automatically extracted from online user feedback. Past studies have found that both App Store Reviews and Tweets contain valuable information related to software evolution and maintenance [11], [12], [13]. Tools that automatically analyse these sources have been proposed [2], [3], [4], [5], [6], [7], [12], [13], [14], [15]. App Store tools have traditionally used Natural Language Processing (NLP) and Machine learning (ML) trained with text features such as, Bag of Words (BOW) and sentiment score [3], [4], [6], [7]. They automatically classify the feedback into pre-defined categories, such as, bug reports or feature requests (e.g., [2]), or extract the specific topic of the feedback (e.g. [6]). Mining requirements from Twitter hasn't been as heavily studied as App Stores. However, in recent years more focus has been given to the platform. Guzman et al. developed classifiers to sort tweets according to their relevance to certain parties, including: technical shakeholders, non-technical shakeholders and the general public [13]. Building on this work, Guzman et al. developed ALERTme to classify, group, and rank requirement-relevant tweets [14]. Such automated analysis has been found to greatly reduce the manual effort in extracting requirements from online feedback [4].

Groen et al. [1] argue that these automated techniques are not yet sufficient to be used on their own, and they propose combing these techniques with crowd sourcing to semi-automate the requirement identification. Automated techniques could be prone to selection bias, but incorporating crowd sourcing could enable a wider range of stakeholder feedback. Tools have been proposed that allow stakeholders to propose new software requirements with the ability for other stakeholders to prioritise these requirements through "upvotes" [16], [17].

Another way of reducing the selection bias of automated techniques is to expand the type of data that is being analysed. Much of the research on automated techniques has been focused on App Store reviews and tweets, but online user feedback exists in other places (e.g. in online discussion forums). Morales-Ramirez et al. [18] described the ontologies of online user feedback for software services and applications, including discussion forums. They described the primary reasons for users to give feedback on forums as reporting a defect, requesting improvements or asking about unclear functionality. Some research has proposed text mining techniques for online forums. Gottipati et al. proposed a technique to automatically identify the most relevant answer in question and answer style forums [19]. In 2014, Morales-Ramirez et al. [20] outlined a methodology for discovering speech acts in online discussions using a rule based (non-Machine Learning) approach. In 2017, building off their previous work, Morales-Ramirez et al. [21] presented a method for the analysis of online discussions within software issue tracking systems. They used a combination of speech acts and sentiment as the features in machine learning algorithms and reported promising performance in classifying feedback as Enhancements, Features or Defects. Stack Overflow, a forum specific to software development expertise, has also been studied. For example, researchers have proposed methods to automatically tag posts [22] and extract software documentation information [23].

However, to the best of our knowledge, no research has investigated software requirement-related content contained specifically in online *product* forums or proposed automated techniques to extract software requirement information from such forums. This study aims to address this gap and investigates the contents of online software product forums, forums where users discuss specific software products. We examine the content of these forums with a focus on potential software requirements and develop new, forum specific, automated classification techniques. We were guided by the techniques developed for analysis of App Store reviews and tweets.

## III. Research Setting

We studied the VLC media player and Firefox web browser forums. Both have a large active community (developers and users) and have been active for many years, making them an ideal research setting. The context of our research setting is described below by considering 5 facets; *product, people, organisation, market* and *processes*, as recommended by Petersen and Wohlin [24].

### A. VLC Media Player

The *product* is VLC, a free, open source, cross-platform multimedia player, initially released in 2001. It supports playing and streaming audio and video files in many different formats. The contributors (*people*) are a diverse and widely distributed collection of volunteers from over 40 countries. The forum has 54 listed site administrators [25]. VLC's development and administration is coordinated by VideoLAN, a French based non-profit *organisation*.

VLC has a large existing *market*. Users have downloaded the application 54.5 million times as of September 2018 [26].

The popularity of VLC means effective product maintenance and evolution is important to meet the diverse needs and expectations of the users. With respect to *process*; the development community meets user needs with regular iterative releases [27]. Bugs and feature requests manually identified in the forum, are often manually transferred to an issue tracking platform. Through our manual analysis, we observed that forum admins often replied to bug reports and feature requests to confirm the new requirement, then prompt the reporter to enter it into the issue tracker. [28]

### *B. Firefox Web Browser*

The *product* is Firefox, a free, open source, cross-platform web browser, initially released in 2004 [29]. Its primary features are tabbed browsing, private browsing, smart bookmarks and a download manager. It allows third party add-ons. The *people* who contribute to Firefox are both volunteers and paid contributors, widely distributed around the world. Firefox's development and administration is coordinated by the Mozilla Foundation, a California-based non-profit *organisation*. [30]

Firefox is one of the worlds most popular browsers and has a large existing *market* of around 500 million yearly active users [31]. With respect to *process*; Firefox has regular, iterative releases and uses Bugzilla to track work for up coming releases [32].

### *C. Data Collection*

We developed custom web-scrappers to automatically mine the user feedback from both forums. We collected all available data at the time of collection. For VLC, 44,300 posts and their replies were collected on May 4, 2018 from two of the forum's topics. We collected 38,000 from the Windows platform section (the largest topic) and 6,300 from the feature request section (the most RE specific topic). On November 24, 2018, 13,000 posts and their replies were collected from the Mozilla Firefox forum.

For each post and its replies we collected the following data; the *title*, the *author(s)*, the *content*, the post *date*, the *number of views* and the *URL*. Additionally, for each user in the VLC forum, we collected the *user level*, a measure of their forum and product experience. Figure 1 shows an example VLC post with the collected data fields labelled.

## IV. Forum Contents

**RQ1:** *What user feedback is contained in the VLC and Firefox product forums?*

### *A. Research Method*

*1) Developing Sentence Classifications:* Starting with the VLC forum, we analysed a random sample of posts and their replies to identify the type of information they contained. Two of the authors inspected each sentence, in isolation from the complete post. For each sentence, either a new classification was created, or an existing classification was assigned. Posts were sampled until no new classifications were observed in three sequential posts (theoretical saturation). In total, 56 VLC posts containing 612 sentences were inspected during this phase.

TABLE I
Intercoder Reliability

| Data set | Initial Agreement | Reconciled Agreement | Cohen's Kappa |
|---|---|---|---|
| VLC | *90.3%* | *95.8%* | *95.1%* |
| Firefox | *92.9%* | *96.3%* | *95.5%* |

Sentence level inspection was chosen, following the approach outlined by Panichella et al. since complete posts can also contain sentences not related to software maintenance [33]. We found that the larger forum posts could contain multiple, disparate, motivations or ideas. Thus, sentences are more likely to contain one coherent idea and, therefore, be more suitable for a precise classification.

This resulted in a set of classifications with a description and example for each, which was used to perform the Manual Content Analysis [34].

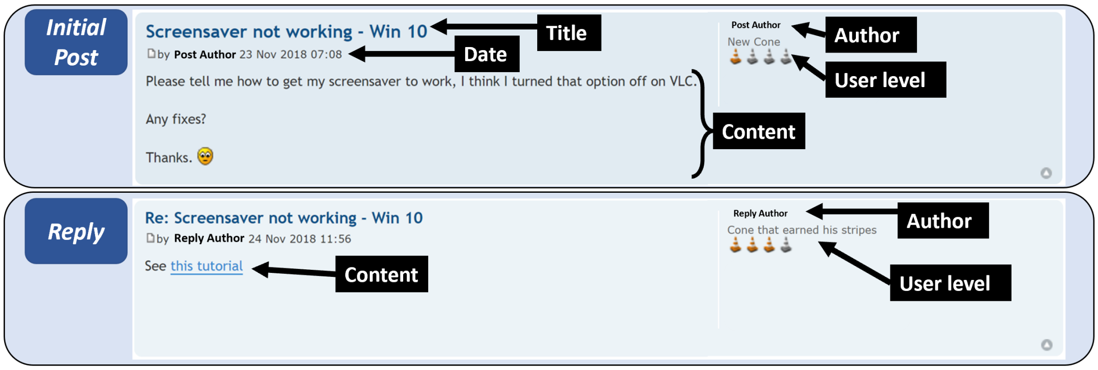
*2) Manual Content Analysis:* After identifying the classification labels, we manually classified a sample of forum sentences on both the VLC and Firefox data. For both data sets, the size of our random sample was determined by calculating the population needed to obtain at least a 95% confidence level with a confidence interval of 10% [35]. This resulted in a manual analysis of 176 VLC posts and 95 Firefox posts (2202 and 1765 sentences, respectively).

Two coders independently manually analysed and classified the same set of sentences using the classification labels and guidelines developed in the previous step. This was performed in multiple rounds, with meetings to discuss and reconcile disagreements between rounds. The intercoder reliability [36] between the two coders, calculated using the ReCal tool [37], shows a high level of agreement (Table I). When the two coders were unable to come to agreement after a discussion, the sentence was discarded, so the analysis for RQ2 and RQ3 includes only sentences that both coders agreed upon. In total, *4.2%* (92) and *3.7%* (65) of the VLC and Firefox sentences (respectively) did not have an agreed classification and were discarded. In the final sets, 2110 VLC and 1700 Firefox sentence classifications were agreed upon and retained. The labelled sentence sets are available in a replication package[1].

### *B. RQ1 Results*

We identified 18 different classifications for the information contained in forum posts and their replies. Table II shows details of each classification, where the label is commonly found (initial post or a reply), how often each label occurs in both data sets (identified through the manual content analysis), and whether past research has previously identified similar classifications in other online user feedback sources. The examples given for each classification are from the VLC and Firefox data. As can be seen, many of the classifications haven't been identified in previous research and may be novel to

[1] https://zenodo.org/record/3340156#.XS-pjegzZPY



Screensaver not working - Win 10

by Post Author 23 Nov 2018 07:08

Please tell me how to get my screensaver to work, I think I turned that option off on VLC.

Any fixes?

Thanks.

Post Author

New Cone

Re: Screensaver not working - Win 10

by Reply Author 24 Nov 2018 11:56

See this tutorial

Reply Author

Cone that earned his stripes

Fig. 1. Example VLC forum post, illustrating data collected.

forums. The RE Relevance column shows how we envision each classification may be useful for eliciting software requirements. While some classifications are labelled non-relevant, these could still be useful, in future work, for pattern recognition in full post analysis, described in detail in section VII.

**Answer to RQ1:** Table II summarises the type of user feedback contained within the forums. We identified 18 different forum sentence classifications. Of these, 11 haven't been identified in previous research.

## V. App Store Tool Assessment

**RQ2:** *Can a tool developed to classify and extract information from App Store reviews be used to classify forum sentences?*

### A. Research Method

Since there is some overlap between the information contained in App Store Reviews and the information we found in forums, we investigated if an existing analysis tool was sufficient in this new domain. For this purpose, we selected ARdoc [2] since it obtains high accuracy in app review classification, is readily available, and functionality includes all required data analysis steps (including pre-processing).

ARdoc classifies sentences into four categories: Problem Discovery, Feature Request, Information Seeking and Information Giving. To enable an evaluation, we mapped the appropriate sentence classes identified in RQ1 to these four high-level categories, shown in Table III. In some cases more than one forum classification is mapped to the high-level ARdoc classes. Any forum classifications that could not be mapped to one of ARdoc's classes, was excluded from this analysis.

We evaluated ARdoc's accuracy on forum sentences (using the sentences labelled in the manual content analysis as the truth set). Accuracy was evaluated using precision, recall and f-measure for each classification type [39]. In the evaluation, we applied ARdoc, which comes pre-trained on App Store reviews [2]. We used the features Panichella et al. [2] found obtained the highest accuracy (sentiment and text structures).

### B. RQ2 results

The accuracy of ARdoc on the forum data is shown in Table IV. ARdoc achieved the highest accuracy for the Information Seeking class in the forum sentences, with *40.9%* (VLC) and *55.1%* (Firefox) f-measure scores. Its worst performance came when identifying Feature Requests, with *18.0%* (VLC) and *4.3%* (Firefox) f-measure.

**Answer to RQ2:** The App Store tool ARdoc, had an average f-measure score over its four classification types of *30.4%*, with a range of *4.3%* to *55.1%*.

## VI. Forum Specific Classifiers

**RQ3:** *Can a classifier that utilises characteristics specific to user forums outperform a classifier developed for App Store reviews?*

### A. Research Method

We used a supervised learning approach to build the new classifiers. We used the classified sentences analysed in RQ1 as the truth sets to train and test the new classifiers.

*1) Algorithm Selection:* We used a binary Naïve Bayes algorithm for our classifiers. Naïve Bayes is a popular binary classifier algorithm, based on Bayesian probability. It has been proven to outperform other algorithms in classifying other types of online user feedback [3] and has been shown to achieve good results with relatively small training data [40].

*2) Sentence Class Selection:* When employing binary classifiers, individual classifiers are needed for each classification type. Past work have found these to outperform a single multiclass classifier [3]. We developed classifiers for all classes where we had at least 100 labelled sentences for that classification in the truth set, to ensure adequate data for training and testing.

Five classifications met the threshold in both the VLC and Firefox truth sets: *Application usage*, *non-informative*, *apparent bug*, *application guidance* and *question on application*. Three additional classifications met the threshold in only the VLC truth set: *Help seeking*, *feature request* and *user setup*. Therefore, eight VLC and five Firefox classifiers were created.

TABLE II
TYPE OF INFORMATION AVAILABLE IN FORUMS: SENTENCE CLASSIFICATIONS

| | Label | Description | Similar To | Post Type | VLC Proportion | Firefox Proportion | RE Relevance |
|---|---|---|---|---|---|---|---|
| 1 | Application usage | The user gives general information about how they are using the application or the area of the application they are interested in. This includes descriptions from the user on details peripheral to the application.<br>*E.g. "VLC and Scandinavian letters"* | [3] | Initial post | 27% | 23% | Context of requirement |
| 2 | Non-informative | Doesn't provide useful information for software maintenance or evolution.<br>*E.g. "Hey there, first of all, i'm new to this forum and english is not my nativ language, so if something is wrong just let me know."* | [11], [2], [38] | Any | 15% | 8% | Non-relevant |
| 3 | Apparent bug | A sentence that indicates the software is malfunctioning.<br>*E.g. "I keep getting an input error after download the update.."* | [11], [2], [3], [38] | Initial post | 12% | 8% | Potential requirement (bug fix) |
| 4 | Application guidance | An explanation of how the software should function/behave.<br>*E.g. "Well those hotkeys loop through the zoom sizes"* | - | Reply | 10% | 33% | Context of requirement |
| 5 | Question on application | A question on the software, including "how to" questions. Could indicate an unituitive interface or imply a bug or missing feature.<br>*E.g. "where is the file or files that have those settings stored?"* | [11] | Initial post | 7% | 7% | Potential requirement (unknown type) |
| 6 | Feature request | Requesting a new feature or describing the way they would like the software to behave.<br>*E.g. "I want to put a time line to some events in the movie"* | [11], [2], [3], [38] | Initial post | 6% | <1% | Potential requirement (new feature) |
| 7 | Help seeking | A **non-specific** request for help, or acknowledgement of help given. Indicating the user is a help seeker. Not asking a direct question.<br>*E.g. "please help me"* | - | Initial post | 6% | 2% | Non-relevant |
| 8 | User setup | Describing the user's software and/or hardware setup. Could include: Software version, OS, hardware, additional software.<br>*E.g. "Im using Windows 7 Ultimate."* | - | Initial post | 5% | 2% | Context of requirement |
| 9 | Question on background | Looking for information not directly related to the operation or behaviour of the main software.<br>*E.g. "CPU is not peaking?"* | - | Any | 3% | <1% | Context of requirement |
| 10 | Attempted solution | User describes an attempted solution that didn't correct the problem. Could be a response to an expert's suggestion.<br>*E.g. "Ive tried older versions of VLC and without success"* | - | Initial post | 3% | 3% | Context of requirement |
| 11 | Requesting more information | An experienced user requests more information to help solve the problem or understand a feature request.<br>*E.g. "Could you upload some sample files?"* | - | Reply | 2% | 6% | Non-relevant |
| 12 | Praise for application | A sentence that praises the application.<br>*E.g. "Overall i like very much vlc"* | [11] | Initial post | <1% | <1% | Indicates sentiment |
| 13 | Dispraise for application | Expresses negative sentiment towards the application and/or its features.<br>*E.g. "I must confess, this is not my player of choice."* | [11] | Initial post | <1% | 3% | Indicates sentiment |
| 14 | Acknowledgement of resolution | Confirms the user's issue has been resolved. Can include details for the solution.<br>*E.g. "Yeah I eventually figured it out"* | - | Initial post | <1% | 4% | Indicates severity |
| 15 | Agreeing with the problem | A new poster says they have the same problem as the original poster.<br>*E.g. "I have a similar problem"* | - | Reply | <1% | <1% | Indicates scale |
| 16 | Limitation confirmation | Confirmation in a reply that the feature being discussed isn't currently implemented.<br>*E.g. "But for the new GUI, you have to be patient."* | - | Reply | <1% | <1% | Validates requirement type |
| 17 | Bug confirmation | Confirmation in a reply that the discussed issue is a software bug.<br>*E.g. "It didn't get fix in 0.9.6, so the problem still exists."* | - | Reply | <1% | <1% | Validates requirement type |
| 18 | Agreeing with the feature request | After a new feature has been proposed, this is a sentence that gives support to that feature.<br>*E.g. "I second that request"* | - | Reply | <1% | <1% | Indicates scale |

TABLE III
MAPPING BETWEEN FORUM AND ARDOC CLASSIFICATIONS

| ARdoc Classes | Mapped Forum Classes |
|---|---|
| Problem Discovery | Apparent bug |
| Feature Request | Feature request |
| Information Seeking | Question on application<br>Help seeking<br>Requesting more information<br>Question on background |
| Information Giving | Application guidance<br>User setup<br>Praise for application<br>Dispraise for application<br>Application usage<br>Attempted solution<br>Acknowledgement of resolution |

TABLE IV
ACCURACY OF ARDOC ON FORUMS

| ARdoc Class | Forum | Recall | Precision | F-Measure |
|---|---|---|---|---|
| Problem discovery | VLC | 0.303 | 0.530 | 0.386 |
|  | Firefox | 0.375 | 0.348 | 0.361 |
| Feature request | VLC | 0.173 | 0.188 | 0.180 |
|  | Firefox | 0.200 | 0.024 | 0.043 |
| Information seeking | VLC | 0.264 | 0.908 | 0.409 |
|  | Firefox | 0.409 | 0.844 | 0.551 |
| Information giving | VLC | 0.151 | 0.324 | 0.206 |
|  | Firefox | 0.184 | 0.749 | 0.296 |

*3) Handling Unbalanced Data:* Since we have a large number of classifications and are developing binary classifiers, for any single class, the number of sentences in that class (called the minority class) are outnumbered by the sentences in all other classes (the majority class). This can cause the classifier to become biased, resulting in most, if not all, sentences being classified as the majority class.

To mitigate this, a classifier can be artificially rebalanced by either over-sampling the minority class or under-sampling the majority class. In under-sampling, some of the majority class sentences are discarded from the training data. A drawback of under-sampling is that potentially useful information is discarded. In over-sampling, additional minority sentences are added to the training data, either by resampling the existing sentences or artificially creating new minority class sentences.

To find the best approach to rebalance the data and optimise performance, we trialled both over- and under-sampling methods and compared the resulting classifiers. Several advanced techniques exist to select or generate the samples that are added or removed, including: SMOTE [41] and ADASYN [42] for over-sampling, or Cluster [43] and Tomek Link [44] for under-sampling. However, simple methods can often match or outperform the more complex techniques [45]. Thus, we used a simple resampling approach for both over- and under-sampling. To over-sample, the minority class sentences, within the training data, were completely resampled iteratively; an extra copy of each sentence in the minority class was added for each over-sampling iteration. To under-sample, samples from the majority class, in the training data, were randomly discarded in iterations of 10% of the total.

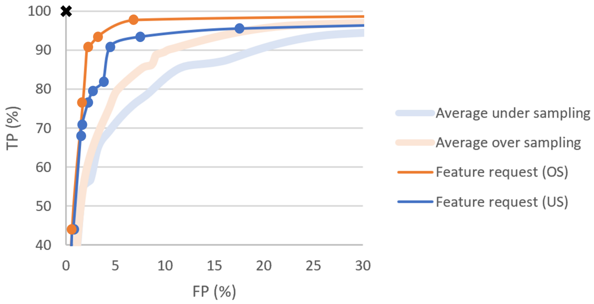


Fig. 2. ROC curves showing Over-Sampling and Under-Sampling performance.

TABLE V
NUMBER OF OVER-SAMPLING ITERATIONS FOR EACH CLASSIFIER.

| Classifier | VLC | Firefox |
|---|---|---|
| Application usage | *3* | *3* |
| Non-informative | *5* | *4* |
| Apparent bug | *3* | *4* |
| Application guidance | *6* | *6* |
| Question on application | *2* | *2* |
| Help seeking | *4* | - |
| Feature request | *4* | - |
| User setup | *5* | - |

To identify the best resampling technique and the ideal number of iterations, we created Receiver Operating Characteristic (ROC) curves [39], which plot the percentage of True Positives (TP) against the percentage of False Positives (FP), for each of our classifiers. The point closest to TP=100 and FP=0 is optimal [39]. Figure 2 shows example ROC curves for the VLC *feature request* classifier as well as the average ROC curves for all classifiers. Each point on the lines, represents one resampling iteration. We found that classifiers using over-sampling consistently outperform those using under-sampling. This is possibly related to the loss of information in discarded samples when under-sampling. For each classifier, we selected the optimal number of over-sampling iterations using the ROC curves (listed in Table V).

*4) Classifier Features:* Potential features were identified by reviewing related literature on App Store Reviews and Tweets and using the domain knowledge gained through the analysis of forum posts in RQ1. To identify features that are meaningful for forums, we trialled various combinations of each of the features.

There are two *word-based features* that have been successfully used in past work. Both of these features consider the presence of one or more words.

*a) Bag of Words (BOW):* a widely used text feature, where a dictionary of all words in the corpus is created. For each sentence, the presence or absence of each word in the dictionary is calculated. As past research has found that pre-processing techniques, like removing stop words or lemmatization, can remove useful information [3], we did not perform pre-processing.

*b) N-grams:* another widely used text feature that groups a continuous sequence of N tokens (words or numbers) in a sentence. Since not all N-grams will be meaningful, we incrementally manually built N-gram lists for each forum sentence class, modelled on the work done by Shi et al. [5]. We used the NLTK python library [46] to generate a list of all N-grams (where N=1-4) and their frequencies in the truth set. Two people manually inspected the lists to create N-gram lists for each sentence classification. For example, the tri-gram "*but when I*", appeared often in *apparent bug* sentences, to indicate unexpected behaviour.

In addition to these two common word-based features, we also used several additional features.

*c) Post Position:* refers to where in the post thread the sentence is located. This feature is novel to forums due to the threaded nature of the posts. Possible values are: (1) Post title, (2) Initial post, (3) Initial poster follow up, (4) 1st Reply, (5) 2nd Reply, etc. Initial poster follow up, captures any reply made by the person who created the initial post. Replies made by someone other than the original poster, are labelled as a numbered reply (based on their order). Table VI shows, for each position (up to the 2nd reply), the percentage each classification makes up of sentences in that position (for the eight classifications we built classifiers for). The "All Positions" column shows the percentage each classification makes up in the complete truth set. For example, 39.6% of all 1st Replies are classified as Application Guidance, while only 10.4% of all sentences in our truth set are Application Guidance. Thus, it can be seen that certain classifications are more likely to appear in certain post positions, therefore the post position could be a useful feature for a classifier. Post titles often contain *Apparent Bugs*, *Feature Requests* or *Questions*. The most likely location for each classification is bolded in Table VI.

*d) Topic:* captures the subforum topic of the post (only available in VLC). The post author selects the topic when creating the post in the forum. The two topics in our dataset were *Windows* and *feature request*.

*e) Verb Tense:* The types of verbs that appear in a sentence can have a correlation to its overall classification [3]. For example, past tense verbs, are used when reporting a previous experience, such as a past issue with the software. Whereas, verbs with a future tense can be used to describe a hypothetical situation, such as a suggested new feature. We used the NLTK part of speech (POS) tagger [47] to tag the types of verbs in each sentence, the verb types are detailed in table VII [48]. When training the classifiers, the simple presence or absence of each verb tag was used as the text feature.

TABLE VI
PERCENTAGE EACH CLASS APPEARS IN EACH POSITION

| | All Positions | Title | Initial Post | Initial Poster Follow Up | 1st Reply | 2nd Reply |
|---|---|---|---|---|---|---|
| Apparent Bug (%) | 12.4 | **31.7** | 16.4 | 10.0 | 1.3 | 7.7 |
| Feature Request (%) | 5.5 | **20.0** | 5.8 | 4.5 | 0.0 | 0.0 |
| Application Guidance (%) | 10.4 | 0.0 | 1.8 | 0.5 | **39.6** | 29.2 |
| Question on Application (%) | 7.4 | **15.2** | 9.5 | 9.7 | 0.0 | 1.5 |
| Help Seeking (%) | 5.6 | 0.0 | **6.9** | 5.9 | 0.4 | 0.0 |
| User Setup (%) | 4.6 | 0.7 | **7.4** | 3.6 | 0.9 | 1.5 |
| Application usage (%) | 27.4 | 30.3 | 27.7 | **33.8** | 26.0 | 24.6 |
| Non-informative (%) | 14.5 | 0.7 | 15.9 | **18.0** | 9.3 | 13.8 |

TABLE VII
TYPES OF VERBS [48]

| Category | Description | Example |
|---|---|---|
| MD | Modal auxiliary (future) | May, should |
| VB | Base form | Think |
| VBZ | 3rd person, present | She thinks |
| VBP | 1st person, present | I think |
| VBD | Past tense | They thought |
| VBN | Past participle | A sunken ship |
| VBG | Present participle | Thinking is fun |

TABLE VIII
VERB TYPE OCCURRENCE RATES IN TRUTH SET

| | MD | VB | VBZ | VBP | VBD | VBN | VBG |
|---|---|---|---|---|---|---|---|
| Application guidance | 1.53 | 1.62 | 1.26 | 1.03 | 0.43 | 1.06 | 0.91 |
| Apparent bug | 0.80 | 0.90 | 1.81 | 1.62 | 1.50 | 1.48 | 1.76 |
| Question on application | 2.23 | 2.16 | 1.64 | 1.17 | 0.70 | 1.20 | 1.48 |
| Feature request | 2.48 | 2.01 | 1.02 | 1.40 | 0.56 | 1.06 | 0.71 |
| Help seeking | 1.48 | 1.01 | 0.34 | 0.35 | 0.04 | 0.53 | 0.56 |
| User setup | 0.05 | 0.08 | 0.62 | 1.00 | 0.98 | 0.83 | 0.95 |
| Application usage | 0.84 | 0.90 | 1.04 | 1.08 | 1.28 | 1.01 | 1.01 |
| Non-informative | 0.11 | 0.23 | 0.12 | 0.24 | 0.23 | 0.03 | 0.17 |

Table VIII shows the occurrence rate of verbs in each sentence class compared to the general rate of the verbs for our truth sets. Future tense verbs (MD) more commonly occur in *feature requests* and *questions on application* sentences. For example, in the *feature request*, "*I would like to see VLC change...*", the MD verb *would* is used to indicate a potential change the user wants. On the other hand, past tense verbs (VBD and VBN) are most common in *apparent bug* sentences. For example, in the *apparent bug*, "*Everything seemed to install correctly but...*", the past tense verb, *seemed*, indicates a previous experience.

*f) Part of Speech Tagging:* Following from the work done with verbs, the other parts of speech tagged by the NLTK's POS tagger, were trialled as text features. Not all the NLTK tags were found to have meaningful correlations to sentence classes, the ones that did are presented in table IX [48].

Table X shows the rate at which the POS tags appear in each sentence class, with respect to the occurrence rate in other sentences. There is a high occurrence of cardinal numbers (CD) in *user setup*, where numbers are commonly used to describe software versions or hardware properties. Adjectives (JJ) are often used in *feature requests*. Posters use adjectives to give more detail about the feature they want, for example in, "*I hope VLC 9 will come with a big phat Media Library*", the adjective *big* is used to add detail about the size of the desired library. Proper nouns (NNP) are very common in *application guidance* and *user setup* sentences. In *application guidance*, proper nouns are used to describe the different components of the software that need attention. For example, in the guidance sentence, "*Setting VLC to DirectX or DirectDraw does fix this problem but...*". *VLC*, *DirectX* and *DirectDraw* are all proper nouns.

TABLE IX
PART OF SPEECH DESCRIPTIONS [48]

| Tag | Description | Example |
|---|---|---|
| CD | Cardinal number | five, three, 13% |
| JJ | Adjective | nice, easy |
| NN | Noun, singular | tiger, chair |
| NNP | Noun, proper singular | Germany, God, Alice |
| RB | Adverb | Extremely, loudly, hard |
| TO | Infinitival to | What to do? |

TABLE X
PART OF SPEECH OCCURRENCE RATES IN TRUTH SET

|  | CD | JJ | NN | NNP | RB | TO |
|---|---|---|---|---|---|---|
| Application guidance | 1.49 | 1.34 | 1.37 | 1.92 | 1.24 | 1.27 |
| Apparent bug | 1.55 | 1.37 | 1.43 | 1.21 | 1.66 | 1.06 |
| Question on application | 1.09 | 1.21 | 1.37 | 0.96 | 1.19 | 1.91 |
| Feature request | 0.44 | 1.67 | 1.50 | 0.91 | 1.14 | 2.02 |
| Help seeking | 0.10 | 0.27 | 0.52 | 0.19 | 0.43 | 0.43 |
| User setup | 4.16 | 0.69 | 0.84 | 1.80 | 0.37 | 0.42 |
| Application usage | 0.95 | 1.06 | 1.00 | 0.97 | 1.05 | 1.11 |
| Non-informative | 0.04 | 0.25 | 0.26 | 0.59 | 0.28 | 0.17 |

*5) Avoiding Over-fitting:* To prevent N-gram over-fitting, we followed the procedure described by Shi et al. [5]. We split the truth set into ten partitions and used the N-grams identified in only the first four partitions. As shown in Figure 3, this allowed us to include most of the identified N-grams for each classification, while leaving six partitions to test for over-fitting. We created classifiers that use only the N-gram features. The ten partitions were grouped into five lots (illustrated in Figure 4). The first two lots are *fitted data*, since the N-grams were sourced from these partitions. The remaining lots are *non-fitted data*. We evaluated the performance of the N-gram only classifier on each of the five lots to ensure that performance was not significantly worse on the non-fitted data.

Table XI shows the f-measures obtained using the N-gram only classifiers for each sentence class comparing the fitted and the non-fitted data. The relative f-measure is the average f-measure of the fitted classifiers (data lots one and two), minus the average f-measure of the non-fitted classifiers (data lots three, four and five). A positive relative f-measure indicates that the fitted classifiers performed better, while a negative score indicates the non-fitted classifiers performed better. The fitted classifiers performed only marginally better overall.

Using only one feature type (N-grams), the performance varied noticeably between the different classifiers (Table XI). Utilising multiple features produced more consistent performance, which is described in *RQ3 results*.

*6) Evaluation:* Performance was measured with the metrics recall, precision and f-measure on the truth sets developed in RQ1 (with over-sampled training data and original test data as explained in Section VI-A.3). We used k-fold cross validation to evaluate our classifiers [39]. K-fold cross validation is a common technique for evaluating performance of a classifier. Five or ten folds (k) are good options according to the variance-bias trade-off [49], [50]. In our evaluation, we used five-fold cross validation.

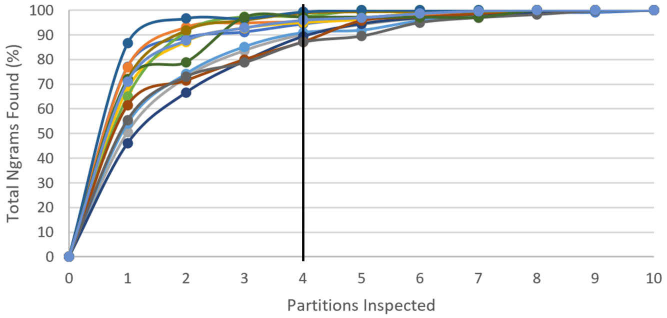


Fig. 3. Ratio of N-grams found in each partition.

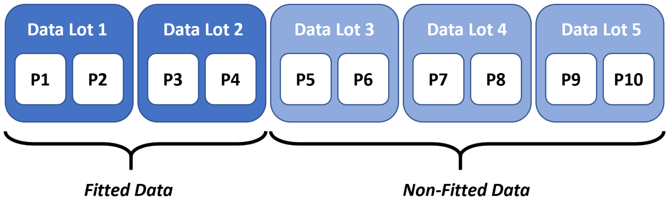


Fig. 4. N-gram over-fitting evaluation approach [5]. Performance of the N-gram classifier is evaluated on each of the five data lots.

### B. RQ3 results

Table XII gives a summary of the average five-fold validation results for each of the binary classifiers. For each classifier, the results are presented for the combination of features that gave the best results.

For all 13 classifiers promising performance was achieved for each metric: recall, precision and f-measure. The average f-measure score, for each best performing classifier, was 78.3%, with the VLC *application usage* classifier being the least accurate at 70.3% and the *application guidance* Firefox classifier the most at 89.8% f-measure.

**Answer to RQ3:** 13 forum specific binary classifiers were created, each targeting an important forum sentence class. Their average f-measure score was 78.3%. This outperformed ARdoc's average f-measure score of *30.4%*, both having been evaluated on the VLC and Firefox truth sets.

## VII. DISCUSSION

We identified a fine-grained classification of information discussed in online software product forums through our analysis of the VLC and Firefox forums (RQ1). We have shown that forums contain significant data that could be useful for software evolution and maintenance, but they contain very different information than has been reported in the more heavily studied feedback platforms, like App Store Reviews and Tweets. Forum posts often start with a user asking for assistance with a specific problem they are having with a software product. Explicit bug reports and feature requests are not as common on forums unless the forum provides a specific subforum for

TABLE XI
N-GRAM OVER-FITTING EVALUATION RESULTS

| Classifier | Fitted Average | Non-fitted Average | Relative F-measure |
|---|---|---|---|
| Application usage | 20.7 | 20.7 | 0.0 |
| Non-informative | 62.3 | 55.1 | 7.2 |
| Apparent bug | 74.9 | 66.9 | 8.0 |
| Application guidance | 59.9 | 58.5 | 1.4 |
| Question on application | 71.7 | 70.3 | 1.4 |
| Help seeking | 71.0 | 80.3 | -9.3 |
| Feature request | 44.2 | 45.1 | -0.9 |
| User setup | 71.2 | 67.2 | 4.0 |
| **Average** | **59.5** | **58.0** | **1.5** |

feature requests (like VLC in our study). Even when bug reports or feature requests do exist in a forum post, they are phrased differently than those in App Store Reviews, and existing tools cannot be accurately used to identify this information (RQ2).

Many (11 of 18) of the classifications we have identified have not been reported in related work on App Store Reviews or Tweets, and may be unique to forums. This is likely due to the unique, discussion-style format of forums. Feedback contained in other online sources, like App Stores, is more unidirectional. More than half of our new classifications (6 of 11) came from the reply fields, which either don't exist or aren't as extensively utilised on other platforms. These replies could be extremely useful in helping to understand new software requirements. Having the more fine-grained classifications for forum feedback could be important to discern accurate information from complete forum posts. Future studies should look at classifying complete forum posts, building on the sentence level classifications. On forums, users often ask "how" to do something, but it is not clear if this is because the software is unintuitive to use, the needed functionality is missing, or a bug exists in the software [18]. Analysis at the post level, by considering patterns across the sentences, can be useful to better understand the full thread. Our fine-grained classifications can enable detailed patterns within the posts to emerge. For example, if a user posts a how-to question ("*question on application*") that is followed by a reply from another user confirming this is related to a bug ("bug confirmation"), the full post could be considered a bug report. Other classifications that appear in the post can be used to understand the context of when the bug occurs (e.g., "*application usage*") or how many users have the same problem ("*agreeing with the problem*"). Table II shows how we envision each classification could be used in analysis of a full post. Further, some of the non-relevant classifications could still be useful in pattern recognition in the full post analysis.

Forum posts are often much longer than feedback on other platforms, users often described detailed context to their problems. Our new fine-grained sentence classifications can help capture this information. One example of a new contextual classification is the commonly seen *user setup* class, where users describe their hardware and/or software platform. Identifying user setup sentences can help understand the context of a software bug or feature request. Future work could use topic modelling to cluster *user setup* sentences into the various platforms, giving insight into the most common platforms used by forum posters seeking help. The platform type could also be correlated against certain bugs to identify software or hardware causes.

In our exploratory study, we have shown it is possible to develop binary classifiers for the various sentence classes that are common in forum posts (RQ3). We present features that can be useful to analyse forum data including post position and topic. We found that manually sourced N-grams (average f-measure of *78.3%*) performed better than a simple Bag of Words (average f-measure of *58.6%*). Bag of Words and N-grams are both commonly used for automatic feedback classification in the related research, often together (e.g. Maalej and Nabil [3] and CLAP [38]). However, to the best of our knowledge no previous work has compared the performance benefits from Bag of Words against manually sourced N-grams. We also described how we employed simple over-sampling techniques to balance the data since having binary classifiers for a large number of classifications results in unbalanced data.

Future work should build on these results to classify full posts. Considering all of the sentences classifications within a post will likely reveal meaningful patterns that will allow for the classification of the full post. Topic modelling can also be used to cluster related posts to give insight into the highest priority requirements.

## VIII. LIMITATIONS AND THREATS TO VALIDITY

In this section, we discuss the internal and external treats to the validity of our research.

*Internal Validity:* A threat to internal validity relates to whether the results really do follow from the data in our study. Specifically, the manual analysis of forum posts is prone to bias. We mitigated this by developing detailed coding guidelines with descriptions of each classification and specific example sentences. We also used two independent coders to classify all sentences. The coders achieved high levels of inter-coder reliability and cases of disagreement were excluded from the truth set.

The sentences excluded due to a lack of agreement between the coders are a possible source of bias. Failure to agree on a classification was generally caused by ambiguous or unclear content. If two trained reviewers couldn't come to a consensus, then it is unlikely that a predictive classifier could give a meaningful classification. Overall, the percentage of discarded sentences was low, at *3.96%*, and should not have a large effect on the validity of our findings.

We showed that ARdoc was not effective in classifying forum data. While other methods and tools exist that could potentially outperform ARdoc, we believe we have demonstrated that forum data is significantly different to that of App Store Reviews and Tweets. Thus, forum specific methods are required.

Additionally, we may have missed useful techniques or features that could improve accuracy of our classifiers. This was an exploratory study, and we don't claim completeness in

TABLE XII
FORUM SPECIFIC CLASSIFIER RESULTS

| Classifier | Forum | Features Used | Average Recall | Average Precision | Average F-Measure |
|---|---|---|---|---|---|
| Apparent bug | *Firefox* | **Ngrams** + post position + verbs | 0.703 | 0.758 | **0.728** |
| | | **BOW** + post position + verbs | 0.515 | 0.524 | 0.517 |
| Question on application | *Firefox* | **Ngrams** + post position + verbs | 0.908 | 0.732 | **0.808** |
| | | **BOW** + post position + verbs | 0.492 | 0.478 | 0.484 |
| Application guidance | *Firefox* | **Ngrams** + post position + verbs | 0.926 | 0.871 | **0.898** |
| | | **BOW** + post position + verbs | 0.798 | 0.903 | 0.847 |
| Application usage | *Firefox* | **Ngrams** + post position + verbs + POS | 0.908 | 0.642 | **0.752** |
| | | **BOW** + post position + verbs + POS | 0.702 | 0.494 | 0.580 |
| Non-informative | *Firefox* | **Ngrams** + post position + verbs + POS | 0.747 | 0.815 | **0.778** |
| | | **BOW** + post position + verbs + POS | 0.941 | 0.581 | 0.718 |
| Apparent bug | *VLC* | **Ngrams** + post position + topic | 0.733 | 0.718 | **0.725** |
| | | **BOW** + post position + topic | 0.584 | 0.484 | 0.528 |
| Question on application | *VLC* | **Ngrams** + post position + topic + verbs | 0.870 | 0.724 | **0.789** |
| | | **BOW** + post position + topic + verbs | 0.454 | 0.424 | 0.436 |
| Feature request | *VLC* | **Ngrams** + post position + topic + verbs | 0.918 | 0.762 | **0.830** |
| | | **BOW** + post position + topic + verbs | 0.555 | 0.414 | 0.474 |
| Application guidance | *VLC* | **Ngrams** + post position + topic + verbs | 0.906 | 0.642 | **0.751** |
| | | **BOW** + post position + topic + verbs | 0.595 | 0.676 | 0.631 |
| Help seeking | *VLC* | **Ngrams** + post position + topic + verbs + POS | 0.736 | 0.844 | **0.783** |
| | | **BOW** + post position + topic + verbs + POS | 0.755 | 0.485 | 0.589 |
| User setup | *VLC* | **Ngrams** + post position + topic + verbs + POS | 0.836 | 0.813 | **0.819** |
| | | **BOW** + post position + topic + verbs + POS | 0.823 | 0.596 | 0.687 |
| Application usage | *VLC* | **Ngrams** + post position + topic + verbs + POS | 0.908 | 0.577 | **0.703** |
| | | **BOW** + post position + topic + verbs + POS | 0.496 | 0.446 | 0.469 |
| Non-informative | *VLC* | **Ngrams** + post position + topic + verbs + POS | 0.810 | 0.827 | **0.816** |
| | | **BOW** + post position + topic + verbs + POS | 0.881 | 0.537 | 0.658 |

terms of classification accuracy. We incorporated features and techniques that have been proven successful in past work and introduced new features and techniques specific to forums. We demonstrated that forums are a promising source of software requirement data.

*External Validity:* A threat to external validity is the potential lack of generalisability of our findings. We mitigated this threat by studying two diverse forums, VLC and Firefox. The applicability of this work to forums in general needs further investigation. Other product forums can have slightly different structures and metadata to that of the forums in this study. Also, different software products will have a variety of features and user bases, meaning the language employed in the posts will be diverse. However, we believe the findings of this research are a promising first step into the previously untapped domain of product forums. The classification definitions are described in general terms, and they were able to be applied to two forums in different domains with high agreement between the independent reviewers. Additionally, the classifier features that achieved the highest performance (N-grams, post position, verbs and POS) can be readily sourced from any product forum.

## IX. CONCLUSION

In this work, we found that software product forums are a rich source of user feedback and discuss new analysis techniques to classify their content. Once extracted, relevant feedback may be useful for eliciting product requirements that are important in guiding the maintenance and evolution of that product. We investigated the types of user feedback contained in the forums of the popular VLC media player and Firefox web browser. We created a catalogue of 18 sentence level classes that are relevant to maintenance and evolution tasks. Some of these classes have been previously described in the related research, however, many have not and may be specific to the forum domain. We have shown evidence that forum data is different than other sources of online feedback, and additional techniques may be needed beyond those used to classify App Store feedback and tweets. Forum posts are threaded discussions, and replies to questions can provide meaningful information. Further, forum posts often contain significant contextual information beyond the problem descriptions.

We described forum specific features and methods that were used to develop binary classifiers for the most common sentence classifications. Forum specific features, like post position and topic, were described and evaluated. Resampling techniques, which are novel in the requirements mining domain, were used to address the unbalanced data inherent from the large number of fine-grained classifications. Promising results were achieved for all the classifiers with f-measure scores ranging from 70.3% to 89.8%. Future work should build on these results to move from sentence-level classification to post-level classification. This study has demonstrated the promise in studying software product forums as a source of software requirements.

## REFERENCES

[1] E. C. Groen, N. Seyff, R. Ali, F. Dalpiaz, J. Doerr, E. Guzman, M. Hosseini, J. Marco, M. Oriol, A. Perini *et al.*, “The crowd in requirements engineering: The landscape and challenges,” *IEEE software*, vol. 34, no. 2, pp. 44–52, 2017.

[2] S. Panichella, A. Di Sorbo, E. Guzman, C. A. Visaggio, G. Canfora, and H. C. Gall, "Ardoc: App reviews development oriented classifier," in *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, ser. FSE 2016. New York, NY, USA: ACM, 2016, pp. 1023–1027.

[3] W. Maalej and H. Nabil, "Bug report, feature request, or simply praise? on automatically classifying app reviews," in *2015 IEEE 23rd International Requirements Engineering Conference (RE)*, Aug. 2015, pp. 116–125.

[4] N. Chen, J. Lin, S. C. H. 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*, ser. ICSE 2014. New York, NY, USA: ACM, 2014, pp. 767–778.

[5] L. Shi, C. Chen, Q. Wang, S. Li, and B. Boehm, "Understanding feature requests by leveraging fuzzy method and linguistic analysis," in *2017 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE)*, vol. 00, Oct. 2017, pp. 440–450.

[6] F. Palomba, P. Salza, A. Ciurumelea, S. Panichella, H. Gall, F. Ferrucci, and A. De Lucia, "Recommending and localizing change requests for mobile apps based on user reviews," in *Proceedings of the 39th International Conference on Software Engineering*, ser. ICSE '17. Piscataway, NJ, USA: IEEE Press, 2017, pp. 106–117.

[7] A. Di Sorbo, S. Panichella, C. V. Alexandru, J. Shimagaki, C. A. Visaggio, G. Canfora, and H. C. Gall, "What would users change in my app? summarizing app reviews for recommending software changes," in *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, ser. FSE 2016. New York, NY, USA: ACM, 2016, pp. 499–510.

[8] "Global 2000: World's larget public companies in 2016," https://www.forbes.com/global2000, accessed: 21 October 2018.

[9] M. Harman, Y. Jia, and Y. Zhang, "App store mining and analysis: Msr for app stores," in *Proceedings of the 9th IEEE Working Conference on Mining Software Repositories*, ser. MSR '12. Piscataway, NJ, USA: IEEE Press, 2012, pp. 108–111.

[10] D. Pagano and B. Brügge, "User involvement in software evolution practice: A case study," in *Proceedings of the 2013 International Conference on Software Engineering*, ser. ICSE '13. Piscataway, NJ, USA: IEEE Press, 2013, pp. 953–962.

[11] D. Pagano and W. Maalej, "User feedback in the appstore: An empirical study," in *2013 21st IEEE International Requirements Engineering Conference (RE)*, July 2013, pp. 125–134.

[12] M. Nayebi, H. Cho, H. Farrahi, and G. Ruhe, "App store mining is not enough," in *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*. IEEE, 2017, pp. 152–154.

[13] E. Guzman, R. Alkadhi, and N. Seyff, "A needle in a haystack: What do twitter users say about software?" in *2016 IEEE 24th International Requirements Engineering Conference (RE)*, Sep. 2016, pp. 96–105.

[14] E. Guzman, M. Ibrahim, and M. Glinz, "A little bird told me: mining tweets for requirements and software evolution," in *2017 IEEE 25th International Requirements Engineering Conference (RE)*. IEEE, 2017, pp. 11–20.

[15] G. Williams and A. Mahmoud, "Mining twitter feeds for software user requirements," in *2017 IEEE 25th International Requirements Engineering Conference (RE)*. IEEE, 2017, pp. 1–10.

[16] R. Z. Moghaddam, B. P. Bailey, and C. Poon, "Ideatracker: an interactive visualization supporting collaboration and consensus building in online interface design discussions," in *IFIP Conference on Human-Computer Interaction*. Springer, 2011, pp. 259–276.

[17] D. Renzel, M. Behrendt, R. Klamma, and M. Jarke, "Requirements bazaar: Social requirements engineering for community-driven innovation," in *2013 21st IEEE International Requirements Engineering Conference (RE)*. IEEE, 2013, pp. 326–327.

[18] I. Morales-Ramirez, A. Perini, and R. S. Guizzardi, "An ontology of online user feedback in software engineering," *Applied Ontology*, vol. 10, no. 3-4, pp. 297–330, 2015.

[19] S. Gottipati, D. Lo, and J. Jiang, "Finding relevant answers in software forums," in *Proceedings of the 2011 26th IEEE/ACM International Conference on Automated Software Engineering*. IEEE Computer Society, 2011, pp. 323–332.

[20] I. Morales-Ramirez and A. Perini, "Discovering speech acts in online discussions: A tool-supported method." in *CAiSE (Forum/Doctoral Consortium)*, 2014, pp. 137–144.

[21] I. Morales-Ramirez, F. M. Kifetew, and A. Perini, "Analysis of online discussions in support of requirements discovery," in *International Conference on Advanced Information Systems Engineering*. Springer, 2017, pp. 159–174.

[22] X. Xia, D. Lo, X. Wang, and B. Zhou, "Tag recommendation in software information sites," in *2013 10th Working Conference on Mining Software Repositories (MSR)*. IEEE, 2013, pp. 287–296.

[23] M. P. Robillard, A. Marcus, C. Treude, G. Bavota, O. Chaparro, N. Ernst, M. A. Gerosa, M. Godfrey, M. Lanza, M. Linares-Vásquez *et al.*, "On-demand developer documentation," in *2017 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 2017, pp. 479–483.

[24] K. Petersen and C. Wohlin, "Context in industrial software engineering research," in *Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement*, ser. ESEM '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 401–404.

[25] "VideoLAN support forum," https://forum.videolan.org/, accessed: September 2018.

[26] "VideoLAN organisation website," https://www.videolan.org/index.html, accessed: September 2018.

[27] "VideoLAN organisation website/ news archive," VideoLAN organisation website, accessed: September 2018.

[28] "The VideoLAN wiki - report bugs," Website link, accessed: 4 July 2019.

[29] "Bbc news," Website link, accessed: 4 July 2019.

[30] "Mozilla foundation website," Website link, accessed: 4 July 2019.

[31] "Firefox public data report," Website link, accessed: 4 July 2019.

[32] "Bug report writing guidelines," Website link, accessed: 4 July 2019.

[33] S. Panichella, A. D. Sorbo, E. Guzman, C. A. Visaggio, G. Canfora, and H. C. Gall, "How can i improve my app? classifying user reviews for software maintenance and evolution," in *2015 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, Sept 2015, pp. 281–290.

[34] K. Krippendorff, *Content analysis: An introduction to its methodology*. Sage publications, 2018.

[35] T. H. Wonnacott and R. J. Wonnacott, *Introductory statistics*. Wiley New York, 1990, vol. 5.

[36] D. Freelon, "ReCal OIR: Ordinal, interval, and ratio intercoder reliability as a web service." *International Journal of Internet Science*, vol. 8, no. 1, pp. 10 – 16, 2013.

[37] "ReCal2: Reliability for 2 coders," http://dfreelon.org/utils/recalfront/recal2/, accessed: 7 November 2018.

[38] L. Villarroel, G. Bavota, B. Russo, R. Oliveto, and M. Di Penta, "Release planning of mobile apps based on user reviews," in *Proceedings of the 38th International Conference on Software Engineering*, ser. ICSE '16. New York, NY, USA: ACM, 2016, pp. 14–24.

[39] O. Maimon and L. Rokach, *Data Mining and Knowledge Discovery Handbook*, O. Maimon and L. Rokach, Eds. Boston, MA: Springer US, 2010.

[40] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python*. O'reilly, 2009.

[41] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *J. Artif. Int. Res.*, vol. 16, no. 1, pp. 321–357, Jun. 2002.

[42] H. He, Y. Bai, E. A. Garcia, and S. Li, "Adasyn: Adaptive synthetic sampling approach for imbalanced learning," in *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, June 2008, pp. 1322–1328.

[43] S.-J. Yen and Y.-S. Lee, "Cluster-based under-sampling approaches for imbalanced data distributions," *Expert Syst. Appl.*, vol. 36, no. 3, pp. 5718–5727, Apr. 2009.

[44] S. Kotsiantis, D. Kanellopoulos, and P. Pintelas, "Handling imbalanced datasets: A review."

[45] E. Liu, J. G. Member, and C. M. Member, "Generative oversampling for mining imbalanced datasets."

[46] "Nltk, python," https://www.nltk.org/, accessed: 21 October 2018.

[47] "Nltk, pos," https://www.nltk.org/book/ch05.html, accessed: 21 October 2018.

[48] "Part of speech descriptions," https://www.clips.uantwerpen.be/pages/mbsp-tags, accessed: 21 October 2018.

[49] S. Fortmann-Roe, "Understanding the bias-variance tradeoff," 2012.

[50] B. Efron, "Estimating the error rate of a prediction rule: improvement on cross-validation," *Journal of the American Statistical Association*, vol. 78, no. 382, pp. 316–331, 1983.