Analysis of Online Discussions in Support of Requirements Discovery

Itzel Morales-Ramirez[1(✉)], Fitsum Meshesha Kifetew[2], and Anna Perini[2]

[1] INFOTEC, Av. San Fernando 37, 14050 Tlalpan, Mexico City, Mexico
itzel.morales@infotec.mx
[2] Software Engineering Research Unit, Fondazione Bruno Kessler, Via Sommarive, 18, 38100 Trento, Italy
{kifetew,perini}@fbk.eu

Abstract. Feedback about software applications and services that end-users express through web-based communication platforms represents an invaluable knowledge source for diverse software engineering tasks, including requirements elicitation. Research work on automated analysis of textual messages in app store reviews, open source software (OSS) mailing-lists and user forums has been rapidly increasing in the last five years. NLP techniques are applied to filter out irrelevant data, text mining and automated classification techniques are then used to classify messages into different categories, such as bug report and feature request. Our research focuses on online discussions that take place in user forums and OSS mailing-lists, and aims at providing automated analysis techniques to discover contained requirements. In this paper, we present a speech-acts based analysis technique, and experimentally evaluate it on a dataset taken from a widely used OSS project.

Keywords: Requirements engineering · Linguistic analysis · Sentiment analysis · Automated classification techniques · Speech-acts

1 Introduction

Social media, together with other web-based communication platforms, including app stores, user forums, mailing-lists, wikis, newsgroups, and blogs are becoming popular means for users of software applications and services to express their quick feedback or engage in online discussions upon their usage experience. The large amount of online data that is accumulated represents an invaluable knowledge source for customer support, software maintenance and evolution, as well as requirements engineering tasks.

The challenges posed by the analysis and exploitation of online data for such purposes is attracting the attention of the software engineering research community. A recent survey [17] on advances and trends in App Store analysis for software engineering points out a huge growth in the number of research works on this topic from 2010 to 2015.

E. Dubois and K. Pohl (Eds.): CAiSE 2017, LNCS 10253, pp. 159–174, 2017.
DOI: 10.1007/978-3-319-59536-8_11

Taking the requirements engineering (RE) perspective, the term *Crowd-based Requirements Engineering (CrowdRE)* [1] has been proposed to indicate the set of concepts, methods and techniques necessary to collect, analyse and manage requirements expressed by members of a crowd of users in the form of online feedback. An ontology for online user feedback has been proposed [20] which characterises the user feedback communication process, the diverse formats in which user feedback can be expressed and the type of information that can be extracted from this feedback. Such ontology helps understand which type of analysis techniques are needed to process online user feedback.

Focusing on the analysis of online textual feedback, Natural Language Processing (NLP) techniques [16] are first applied to filter out irrelevant data. Text mining, such as sentiment analysis and topic modelling, and automated classification techniques are then used to identify feedback that can fall into different categories, such as *bug report* and *feature request* [15,23], and serve as input to release planning tools at support of app developers to identify which new feature to include in the next release [26]. Manual analysis and supervised machine learning techniques are applied to investigate tweets related to software applications so as to characterise their relevance to software engineers and to non-technical stakeholders [11].

Our research focuses on feedback provided through online discussions related to software applications, such as those that take place in issue tracking systems and open source software (OSS) mailing-lists. According to user feedback ontology [20], they can be considered as explicit, directed feedback in which the sender reveals her intention and can affect the receivers' attitude about a subject, through the specific *speech-acts* [24] she uses in her comments. That is the structure of the sentences through which this feedback is expressed and can provide useful information about the intention of the user who expressed it and helps better interpret the user's experience that generated it. In order to capture such users' intentions, other text mining techniques, different from sentiment analysis and topic modelling, needs to be exploited.

In a previous work [18] we explored the applicability of the Speech-act theory [24] to develop a technique for supporting the analysis of OSS mailing-list discussions. *Speech-act* analysis has already been applied to the analysis of online discussions, for example, to investigate most frequent intentions expressed in status messages by users of social networks [3], and in the online teaching domain to understand students' intentions expressed in their queries to teachers [9], or during discussions with peers [14]. In these works, the *speech-acts* that are used in conversations about the specific domain under consideration are first identified, and then used during manual annotation of the conversations, which is performed by independent annotators.

In this paper, we present a revised version of our *speech-act* based analysis technique that we contrast and combine with sentiment analysis when processing online feedback about software applications with the purpose of identifying relevant information for discovering requirements. Specifically, we consider the following research questions:

RQ1: What are the speech-acts expressed in online discussions that may lead to discover requirements?
RQ2: Can the speech-acts be used as a parameter[1] *to classify defect reports, and feature or enhancement requests?*

We answer these questions by analysing online discussions taken from the issue tracking system of the Apache OpenOffice (AOO) community.

The contributions of our research work can be summarised as follows. We provide: (a) insights of a new analysis technique that considers *speech-acts*, in contrast and in combination with sentiment analysis; and (b) interpretation of the *speech-acts* expressed in discussions about software. In addition we provide (c) a new dataset of 6568 threads (40872 comments) from the issue tracking system of AOO, wherein each sentence has been annotated with an intention.

The rest of the paper is organised as follows. In Sect. 2 we present two motivating scenarios that help illustrate how we intend to apply the proposed techniques in a practical setting. In Sect. 3, we first give a brief background of the natural language processing analysis techniques that we rely on, we describe the dataset we used in our linguistic analysis, then state the research questions and the approach we followed to answer them. The analyses are detailed in Sect. 4 and the main findings are discussed is Sect. 5. In Sect. 6 we describe main related work. Conclusion and future work are given in Sect. 7.

2 Motivating Scenarios

We consider two scenarios where online discussions are present in the context of software maintenance and evolution. The first one concerns the analysis of comments from issue tracking systems in large OSS projects, like OpenOffice. In this scenario we imagine that an analyst has to analyse tens of messages per day, approximately. The second scenario is taken from a real case study that is considered in the context of the SUPERSEDE[2] project, which aims at creating an entire set of integrated methods and tools to enable a feedback-driven approach to software lifecycle management. This case study concerns a small company, where the help-desk responsible, a highly experienced person, manages tens of user feedback per months, which are collected through different channels, building an effective bridge between end users and product developers.

2.1 Scenario 1: Open Source Project, User Feedback Analysis for Software Maintenance and Evolution

In OSS communities, collaborators work in a distributed way and the number of contributors can be up to 100,000[3]. There are many developers, analysts and

[1] We use the term parameter as synonym of feature in a machine learning classification approach to avoid confusion with the term feature, which indicates a software application requirement or property, whenever necessary.
[2] http://www.supersede.eu.
[3] http://www.slideshare.net/blackducksoftware/open-source-by-the-numbers.

users that are involved and interested in maintaining and evolving software such as web servers, IDEs, productivity suites, etc. Some developers that work for companies are contributors of open source since companies use open source as core, such as operating systems, databases and development tools[4]. The contributors convey their concerns, for example any bug, new features, or they suggest modifications. There is a continuous exchange of messages that must be read, analysed, replied and considered for making a decision and put it into action. We refer to some of these messages as user feedback, specifically those messages that come from users of the software and that must be read by the OpenOffice volunteers.

The user feedback is collected through e-mail, forum, or issue tracking systems. This implies an asynchronous communication developed as a chain of written messages in natural language (e.g. "One of my spreadsheets is no longer showing in my documents"). Sometimes there are messages expressing a praise that motivates volunteers to continue their work to improving the OSS, other times there are only complaints. The important point is that OSS volunteers form a special kind of community that achieves the maintenance and evolution of software in a distributed setting with a continuous online communication.

The high amount of feedback sent by users is processed manually by any available volunteer, but sometimes it takes the involvement of other specialists to solve the issue that are better suited to deal with such an information. The idea of providing a tool to support the filtering of feedback and redirect it to the right role (analyst or developer) may be crucial to save time.

2.2 Scenario 2: SUPERSEDE User Feedback Analysis for Software Maintenance and Evolution

SEnerCon is a partner of the SUPERSEDE project. It is a small company with more than 25 years of experience in the domain of energy efficiency management, which employs about 15 software developers and engineers. SEnerCon provides several web applications including: (1) an application that enables end-users' (house owners) to monitor and analyse their energy consumption, called interactive Energy Saving Account - iESA; and (2) applications that guide and advise the end-users on how to save energy in every day life through behavioural or technical changes.

The iESA application counts thousands of users. Most of the features of the application are free, and the only obligation for users is to register and accept that their data (including usage logs) are used and analysed by the company upon anonymisation.

SEnerCon collects hundreds of user feedback per month gathered through five main channels, namely contact form, e-mail, hotline, forum and app stores. The end-users express their feedback using natural language text. According to the helpdesk, the end users may prefer to use a forum rather than a dedicated feedback channel because "they might perceive to get a broader audience in

[4] http://www.slideshare.net/blackducksoftware/2016-future-of-open-source-survey-results.

the forum and make them think that SEnerCon will pay more attention to a feedback elaborated through discussions in a forum".

The analysis of end-user feedback is performed by the help-desk who can ask for clarifications, this analysis can motivate the inclusion of new tickets in the issue tracking system that is used by the development team to keep record of pending issues, which can be addressed during maintenance or product evolution.

Specifically, the help-desk reads the user comments from the forum, try to understand if it is an information request he may answer directly, a complaint for something not working properly or if they contain suggestions for new features. This task seems not so effort demanding considering that feedback arrives at a rate of 2 to 5 per day, but it seems to require a strong experience to manage user feedback about the product in order to make it in an efficient and effective way. This becomes evident when the help-desk takes a holiday, and it is "almost impossible to find some one who can perform his task at an acceptable level".

Having a tool that *(i)* suggests if a comment could be motivated by a bug to fix; a new feature to be considered for next release, or new ideas to be further analysed; *(ii)* indicates which bug/feature requests are more important; and *(iii)* indicates if a request is related to previously addressed requests, will provide great help to enable a user-feedback driven evolution of the software product.

3 Speech Act Based Analysis of Online Discussion

3.1 Background

The use of NLP techniques in software engineering is quite popular due to the use of natural language text in key artefacts, such as requirements, test cases, and comments in code, and, more generally, to the fact that most of the information handled by practitioners is textual-based, as highlighted in the technical briefing by Arnaoudova et al. [2]. In our work we focus on artefacts that contain conversations between stakeholders, such as between end-users and the help-desk team, or between users and developers, thus the dialogue structure, which rests on dialogue acts, can provide relevant information. Dialog acts can be studied using the *speech-act* theory of Searle [24]. In a nutshell, *speech-acts* theory claims that when a person says something she/he attempts to communicate certain things to the addressee, which affect either their believes and/or their behaviour. Dialog acts represent the meaning of an utterance at the level of illocutionary force [25][5].

The interesting part of understanding dialog acts or *speech-act* is that they constitute the basis of communication and the application can be, for example, a meeting summariser needs to keep track of who said what to whom, and a conversational agent needs to know whether it was asked a question or ordered to do something [22,25].

In the work of Novielli and Strapparava [22] the approach can be easily extended to other languages by simply redefining the seeds (lexical cues) used in the definition of the dialogue act profiles and by using a POS-tagger and

[5] An illocutionary force refers to the pronouncing of a statement with an intention.

a morphological analyser trained on the target language. While the work of Stolcke et al. [25] developed a probabilistic approach to dialogue act modelling for conversational speech and tested on a large speech corpus (both works use the Switchboard corpus of human-human conversational telephone speech [10]).

The NLP framework used in this work is GATE (General Architecture for Text Engineering) which is a Java suite of tools [6], developed by the University of Sheffield in UK, for building and deploying software components to process human language. GATE can support a wide range of NLP tasks for Information Extraction (IE). IE refers to the extraction of relevant information from unstructured text, such as entities and relationships between them, thus providing facts to feed a knowledge base [5]. GATE is widely used both in research and application work in different fields (e.g. cancer research, web mining, law). This tool is composed of three main components for performing language processing tasks, namely the *Language Resources* component that represents entities such as lexicons, corpora or ontologies; the *Processing Resources* component, which contains a library of executable procedures, such as parsers, generators or ngram modellers; and the *Visual Resources* component that provides visualisation and editing functions that are used in GUIs.

3.2 Dataset: OpenOffice Online Discussions

The collection of the data was through a request done to the OO community via the mailing list dev@openoffice.apache.org on August 1st, 2013. We asked for a dump of the dataset corresponding to certain parameters such as: comments of the threads referring to the Writer application, issue type should be *Defect* and *Feature* or *Enhancement*, and the final parameter was to obtain the discussions of the last year (i.e. between the years 2012–2013). The provided data was a total of 6568 threads in the format of XML files, we parsed these files and stored them in a MySql database for a better manipulation of the information contained in the comments. Each thread contains at least one comment, but some can have more than one hundred comments. The total number of comments is 40872 and we divided each comment into sentences in order to store each one in the database for its later analysis. During the parsing of sentences we removed text enclosed into HTML elements *<* and *>* considering the text as noise; we applied some regex patterns to remove dates, identify links and unify them into a unique codification (i.e. http://www.); we eliminated common contractions and file extensions. Smiley faces or sad faces were replaced by the words SMILEY and SAD, respectively. Other elements, for example *–, =, -, " ..* are removed from the text; or double elements replaced for one, for example *?? to ?*.

We use this data because the members are very active in collaborating and communicating through written messages. Moreover, we are exploring a new type of dataset instead of the commonly used from App stores. In addition to this, the data refers to online discussions (threads) whose first comment has the characteristic of being already labelled by members of the community.

Besides this, we believe the data contains *speech-acts* which are more complex than sentiments and that can provide extra information useful to developers or

requirements analysts. Another important characteristic of the dataset is that it has other properties such as *status* of the issue (e.g. confirmed, unconfirmed), *priority* (e.g. P1-highest priority), *severity* (e.g. blocker, critical) that together constitute a combined property called *importance*.

3.3 Approach

We have two research questions to explore a new dimension of analysis that we call *speech-acts* based analysis.

- *RQ1 What are the speech-acts expressed in online discussions that may lead to discover requirements?*
- *RQ2 Can the speech-acts be used as parameters to classify defect reports, and feature or enhancement requests?*

To answer *RQ1* we use NLP tools such as the Stanford CoreNLP[6], GATE framework[7] and SentiWordNet[8]. We used the Stanford CoreNLP to break the comments into sentences and to get the overall sentiment per sentence. The GATE framework has been used to perform the analysis of *speech-acts* on each sentence and determine the category to which it may belong. The categories of *speech-acts* have been described in [19] and the classification is determined by applying some lexico-syntactic rules, which includes a list of keywords. Finally, SentiStrength[9] was used to evaluate the sentiment of the nouns, verbs and adjectives contained in each sentence.

We obtain the relative frequency of the sentences that are classified into *speech-acts* and plot their distribution. We are interested in knowing the distribution of the *speech-acts* in the comments labelled as *Defect* and *Feature* or *Enhancement*. This would let us know which types of *speech-acts* are highly frequent in Defect, Feature or Enhancement comments.

We answer *RQ2* by considering *speech-acts* expressions and verbs as features of machine learning algorithms, along with the sentiment of verbs, nouns, adjectives and the overall sentiment of a sentence. Specifically, we considered elements such as the number of times *speech-acts* appear in a sentence, the number of times a verb related to specific *speech-acts* appear in a sentence, the number of positive or negative adjectives, as well as the verbs with an associated sentiment. We formulated 41 features in total to be used by machine learning algorithms and applied the well-known metrics Precision and Recall.

Moreover, we investigate the correlation of the type of *speech-acts* and the importance of the comments. The importance is constituted by the *priority* (e.g. P1-highest priority), and *severity* (e.g. blocker, critical). For example, the priority P1 blocker is considered the highest in importance. We then determine the significance of associating *speech-acts* and the importance of comments by applying the chi-squared test.

[6] http://stanfordnlp.github.io/CoreNLP/.
[7] https://gate.ac.uk/.
[8] http://sentiwordnet.isti.cnr.it/.
[9] http://sentistrength.wlv.ac.uk/.

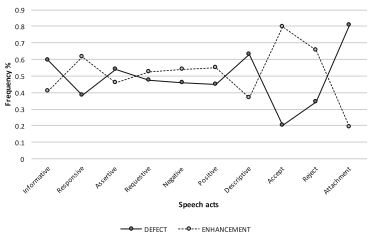

Fig. 1. *Speech-acts* annotated with GATE

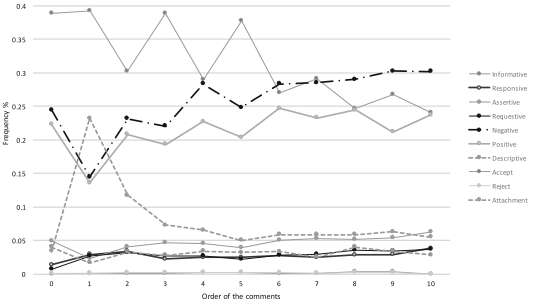

Fig. 2. Distribution of *speech-acts* by the order of the comments, for defect reports

4 Analysis Results

RQ1. What are the *speech-acts* expressed in online discussions that may lead to discover requirements?
Figure 1 reports the distribution of *speech-acts* that have been found in the online discussions of OO[10], using the GATE tool to annotate them. The y-axis shows the percentage of *speech-acts* and the x-axis shows the types of *speech-acts*. We merged the classes Feature and Enhancement, the other class corresponds to Defect.

There is a mirror effect of the lines and there are interesting peaks for each one of the classes. On one hand, we can see that for the class Enhancement the *speech-acts Responsive*, *Negative*, *Positive*, *Accept* and *Reject* have more presence in the discussions. This could mean that members tend to give more responses and engage in the discussions through accepting (80%) or rejecting (66%) new ideas. We also can assume that the participation increases and sentiments are exposed more frequently when the members discuss an Enhancement and that the description of the new feature or enhancement is fully described in the discussion if we consider that there are discussions with more than 100 comments[11].

On the other hand, the discussions regarding a Defect contain the *speech-acts Informative*, *Assertive*, *Descriptive* and *Attachment*. The interpretation we give to this is that it is mandatory to describe (65%) and give details (i.e., attachments with 80%) when there is a problem such that there is no need of further discussion and the Assertive *speech-acts* also emphasises it, for example by stating "I have a problem with...", that is why the *speech-acts* Accept and Reject are not so relevant when discussing a Defect. We also see the presence of sentiments, however they do not make a big difference between positive or negative sentiments.

Figure 2 shows a plot of the different types of *speech-acts* but along the order of how the comments were posted for the Defect reports. This is for all the

[10] The dataset is available at http://se.fbk.eu/technologies/speechactsanalysis.
[11] For instance https://bz.apache.org/ooo/show_bug.cgi?id=3395.

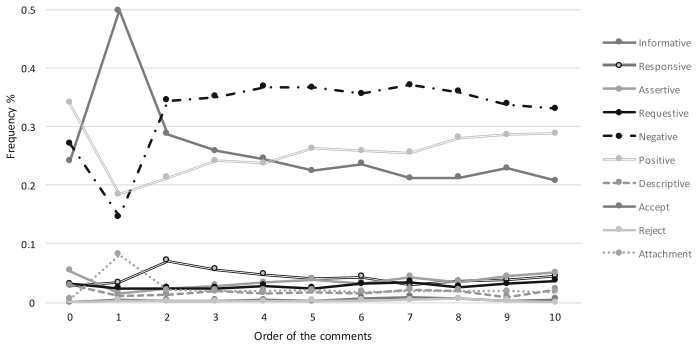

Fig. 3. Distribution of *speech-acts* by the order of the comments, for enhancement and feature requests

threads and we only took a snapshot until the 10th comment. The y-axis displays the percentage of *speech-acts* annotated and the x-axis corresponds to the order of the comments. It is worth to notice how the initial comment (order equal to zero) contains more Informative *speech-acts*, then it decreases but showing some peaks. While the immediate comment after the initial one shows a peak, with a distribution of 23% Attachment and 39% Informative. After comment number 1 the *speech-act* Attachment decreases. The sentiments Negative and Positive have a high presence in the first comment >20% but after that they decrease and later the Negative sentiment goes up and overpasses the Positive and Informative. The other *speech-acts* remain below 5% approximately.

On the other side, Fig. 3 displays the distribution of *speech-acts* for Enhancement and Feature requests, where the Positive *speech-act* has the highest percentage of 35%, followed by Negative and Informative *speech-acts*. The following comments show three interesting peaks, where the distribution is 50% for the Informative *speech-act*, and Negative and Positive sentiments are below 20%. Then, starting from comment 2 the Negative sentiment overpasses the others and keeps a distribution above 30%.

RQ2. Can the *speech-acts* be used as parameters to classify defect reports, and feature or enhancement requests?
We identified 41 features, 7 of them correspond to the *speech-acts*, 13 correspond to the sentiment of a sentence (number of positive and negative verbs, nouns, adjectives, their sentiment score, and the overall sentiment of a sentence). The rest 21 features are, for instance, number of verbs, nouns, adjectives, sentence length, number of question marks, exclamation marks, number of brackets, the number of identified code lines, among others. We trained three machine learning algorithms (Random Forest-RF, J48, SMO) in Weka[12] using these features for classifying the comments labelled as Feature, Enhancement and Defect. We performed a sensitivity analysis by excluding the *speech-acts* features or the

[12] http://www.cs.waikato.ac.nz/ml/weka/.

Table 1. Merged enhancement and feature, 41 features

	RF			J48			SMO		
	P	R	F-M	P	R	F-M	P	R	F-M
Enhancement	.70	.66	**.68**	.66	.62	.64	.61	.77	.68
Defect	.72	.76	**.74**	.68	.72	.70	.74	.56	.64

Table 2. Merged enhancement and feature, 28 features (no sentiment)

	RF			J48			SMO		
	P	R	F-M	P	R	F-M	P	R	F-M
Enhancement	.66	.66	.66	.69	.64	.66	.59	.81	**.68**
Defect	.70	.70	.70	.70	.75	**.72**	.75	.50	.60

Table 3. Merged enhancement and feature, 34 features (no speech acts)

	RF			J48			SMO		
	P	R	F-M	P	R	F-M	P	R	F-M
Enhancement	.68	.64	**.67**	.63	.60	.61	.58	.82	.68
Defect	.70	.74	**.73**	.66	.70	.68	.75	.48	.59

Table 4. 41 features

	RF			J48			SMO		
	P	R	F-M	P	R	F-M	P	R	F-M
Feature	.50	.02	.04	.08	.04	.06	0	0	0
Enhancement	.65	.62	**.63**	.59	.57	.58	.59	.66	.62
Defect	.70	.78	**.74**	.67	.71	.69	.69	.68	.69

sentiment features. Moreover, we merged the comments labelled as feature and enhancement into one class, i.e. Enhancement.

We see in Table 1 that the Random Forest algorithm gives better results compared to J48 and SMO, but it is important to notice that comments labelled as Feature and Enhancement have been merged into Enhancement. The F-Measure (F-M) for Enhancement is .68 and for Defect is .74. Moreover, we use the 41 features, i.e. the *speech-acts* and sentiment features. In Table 2 we see that J48 performs better for Defect comments with a F-M of .72 while the SMO performs better for Enhancement comments with .68, but in this case we removed the features related to sentiments, resulting in 28 features.

Table 3 presents the results of merging Enhancement and Feature and removing the features related to the *speech-acts*. In this case we train the three algorithms with 34 features and the best results for Defect and Enhancement are F-M .67 and .73, respectively. Although, SMO performs better for Enhancement comments with F-M of .68.

In Table 4 we show the results of using the 41 features without merging the Enhancement and Feature comments. The RF algorithm performs better but only for Enhancement with F-Measure .63 and .74, while .04 for Feature. This is mainly due to the small number of comments in the dataset labelled as Feature.

Table 5 shows the results of using the 28 features of *speech-acts* and the best results are given by the J48 algorithm. Again the only good F-M results are for Enhancement .61 and Defect .73. Finally, in Table 6 we apply the 34 features, dropping those related to *speech-acts*. For the Enhancement comments the best F-M result is .62 given by the SMO and for Defect .72.

Table 5. 28 features (no sentiment)

	RF			J48			SMO		
	P	R	F-M	P	R	F-M	P	R	F-M
Feature	.14	.02	.03	.08	.01	.02	0	0	0
Enhancement	.61	.61	.61	.63	.60	**.61**	.58	.64	.60
Defect	.70	.74	.71	.69	.76	**.73**	.68	.66	.67

Table 6. 34 features (no speech acts)

	RF			J48			SMO		
	P	R	F-M	P	R	F-M	P	R	F-M
Feature	.33	.02	.03	.08	.05	.06	0	0	0
Enhancement	.63	.58	.60	.55	.56	.58	.53	.74	**.62**
Defect	.70	.77	**.72**	.65	.70	.67	.71	.54	.62

The chi-squared test to determine the significance of associating *speech-acts* and categories of issues (i.e. Defect, Enhancement or Feature) to the level of *importance* assigned to the comment has been applied for the Defect comments and the merged comments classified as Enhancement or Feature. For Defect comments we have a p-value $<2.2e{-}16$ and for Enhancement and Feature a p-value $= 1.65e{-}05$, indicating that both results are significant and that *speech-acts* could be used to determine the importance associated with a comment.

5 Discussion

Regarding the results of our first research question, we can observe that there is a likelihood that certain types of *speech-acts* are more used when reporting a Defect than an Enhancement issue. We found for instance that 80% of the time the *speech-act* Attachment is expressed for reporting Defects. When an Enhancement or Feature has been reported, between 80% and 60% of the time there is a discussion about accepting or rejecting the ideas exposed by the participants in the discussion.

On one side, although the percentage of the *speech-acts* Responsive, Requestive and Positive is not higher in Enhancement comments than in Defects, we believe that the combination of Requestive and Positive *speech-acts* give a hint of a possible requirement. On the other side the *speech-acts* Informative, Assertive, and Descriptive are more representative for Defect issues.

Besides the combination of *speech-acts*, we saw that there is a trend of such *speech-acts* along the order of the comments. For example, in Defect issues the first comment (#0) contains more Informative, Negative and Positive speech acts, while in the second comment (#1) the Attachment is provided along with the Informative *speech-act*. When it comes to the Enhancement and Feature comments, in the first one the Positive, Negative and Informative *speech-acts* are more present. But the presence of Negative and Positive *speech-acts* drops, while the Informative increases and the Attachment appears as well.

When we use the *speech-acts* as features for training algorithms we notice that by using the *speech-acts* along with sentiment features we get good results when merging Enhancement and Feature comments, instead of just using either *speech-acts* or sentiment features.

Although there is a minimal difference of 1% when we reduce the features by dropping the *speech-acts*, we also know that the amount of Feature comments is not enough for getting good results. Regarding the result of the chi-squared test, of associating *speech-acts* and the importance, we need to investigate further how to recognise which are the types of *speech-acts* that are present for high importance of Defect reports and the same for Enhancement and Feature reports.

5.1 Threats to Validity

Here we discuss the main threats to validity [27]. *Conclusion validity* threats concern issues that affect the ability to draw the correct conclusion on the observed

phenomenon. The results reported in this work give a positive exploitation of *speech-acts*, but we know the dataset must be extended to other years.

Internal validity threats concern the possible confounding elements that may hinder a well performed experiment. Our *speech-acts* based analysis rests on rules which are not extensive but can be improved and the dataset has been already labelled by the OpenOffice community, which means there is not biased in assigning the categories of Defect, Feature or Enhancement.

Construct validity threats concern the relationship between theory and observation. So far there is no theory explaining any correlation between *speech-acts* and categories of issues reported in online discussions related to software applications. Our method of analysis represents a first hypothesis of such relationship.

External validity threats concern extending the validity of observations outside the context. We need to apply the method on other datasets such as the SEnerCon scenario and compare with the results we have obtained until now.

6 Related Works

Research on automated analysis techniques of online discussions at support of requirements engineering tasks has increased significantly, especially in the last five years [17]. The main objective of the proposed analysis techniques is the classification of user comments into bug reports, feature requests, or polarity of sentiments.

For instance, Fang and Zhan [8] apply sentiment analysis on a dataset of 5.1 million product reviews from Amazon. Their approach consists of removing all subjective content (i.e. all sentiment sentences containing at least one positive or negative word). The sentences are tokenised and POS tagged in order to identify adjectives, adverbs, and verbs which are words that mainly convey sentiment.

Worth mentioning is the work of Carreño and Winbladh [4] that aims at analysing comments from users of software applications. Information extraction techniques and topic modelling are exploited to automatically extract topics, and to provide requirements engineers with a user feedback report, which will support them in identifying candidate new/changed requirements.

The research work by Keertipati et al. [13] uses four attributes to be exploited to do the prioritisation, i.e. frequency of a feature, rating, emotions and deontics. They propose three prioritisation approaches: (1) individual attribute-based - when ranking features based on their frequency, the ratings are not considered; (2) weighted approach - enables the combination of two or more attributes in the prioritisation; (3) regression-based approach and data-driven approach to examine influential variables for determining the severity of reviews.

The work of Guzman et al. [12] presents an approach called DIVERSE that aims at recognising the diversity of opinions on a set of App reviews. Moreover, this approach also helps developers and analysts recognise conflicting opinions regarding a feature. The evaluation is performed on a dataset of 170,829 App reviews and a truth set of 2800 manually labelled reviews.

Di Sorbo et al. [7] propose an approach that consists of a two level classification model which considers the users' intentions and review topic of app reviews. Moreover, they propose a summariser called SURF that automatically extracts topics, classifies the intention and group sentences covering the extracted topics for recommending software changes.

Manual analysis and supervised machine learning techniques are applied to investigate the usage and content of about 11 million tweets related to 22 software applications, and their relevance to software engineers and to non-technical stakeholders (such as other users) [11]. Among the main findings, the proportion of information that is relevant for software engineers is small compared to the volume of tweets, thus motivating the development of automated filtering and classification techniques for the exploitation of such online data for requirements engineering purposes.

Finally, a recent work [21] reports a study about the source of requirements in OSS projects by analysing mailing-list discussions along different dimensions including role of participants (pheripherical vs. core participants), and sentiment of end-users. Classification techniques (specifically Naive Bayes algorithm), and sentiment analysis are exploited.

Besides NLP techniques to filter out irrelevant information, these works combine text mining, sentiment analysis and classification techniques. To our knowledge none exploit models of the online discussion which relies on speech-acts. While this may be less relevant for short user feedback, such as tweets and app reviews, in our opinion for online discussions, such as user forum and mailing-list threads, understanding discussants' intentions, which are revealed by the speech-acts they use, could improve feedback classification.

7 Conclusion

In this paper we presented a method for the analysis of online discussions about software products that take place in issue tracking systems. This method aims at supporting a user-feedback driven software evolution approach. The method uses a linguistic technique called *speech-acts* based analysis. To characterise it we proposed two research questions that we answered by applying it to a dataset of the OpenOffice community, which contains 40872 comments that were extracted from the issue tracking system.

The first question investigated the correlation between the use of certain *speech-acts* and categories of issues (e.g. Enhancement, Defect) regarding the software the online discussions are arguing about. With the second question we understand if *speech-acts* used in these comments may provide a relevant parameter for classifying online discussions into Defects, or Feature and Enhancement requests.

We found that there is an association between types of speech acts (e.g. Informative, Responsive, Requestive, etc.) and categories of issues (e.g. Enhancement, Defect). We investigated the distribution of *speech-acts* for the first ten comments for Defect and Enhancement, and identified common patterns on how

conversations about these issues are started and evolve throughout the discussion threads.

We used the *speech-acts* and the sentiment as parameters for training three machine learning algorithms (Random Forest, J48 and SMO) and classify comments into Enhancement, Feature and Defect. Considering the merged Enhancement and Features comments category, resulted in a F-Measure of 0.68 for the merged category and 0.74 for Defect.

In future work, we plan to apply our analysis on user feedback from real world software application, such as SEnerCon's iESA (see Sect. 2), and validate findings with the company's development team. In particular will investigate if our approach could help the team identify relevant feedback that got ignored by manual analysis. Furthermore, we plan to investigate the correlation between *speech-acts* and characteristics of the OpenOffice dataset such as importance, priority and severity.

Acknowledgement. We thank Rob Weir for providing the OpenOffice dataset. This work is a result of the SUPERSEDE project, funded by the H2020 EU Framework Programme under agreement number 644018. The first author is partially funded by INFOTEC under the project "081-022-00-FORTALECIMIENTO E INVERSIÓN".

References

1. Adam, S., Seyff, N., Perini, A., Metzger, A.: Message from the chairs. In: 2015 IEEE 1st International Workshop on Crowd-Based Requirements Engineering (CrowdRE), pp. iii–iv, August 2015. doi:10.1109/CrowdRE.2015.7367580
2. Arnaoudova, V., Haiduc, S., Marcus, A., Antoniol, G.: The use of text retrieval and natural language processing in software engineering. In: Proceedings of the 37th, ICSE 2015, pp. 949–950. IEEE Press (2015)
3. Caleb, C., Schrock, D., Dauterman, P.: Speech act analysis within social network sites' status messages. In: 59th International Communication Association Conference, vol. 20, May 2009
4. Carreño, L.V.G., Winbladh, K.: Analysis of user comments: an approach for software requirements evolution. In: Notkin, D., Cheng, B.H.C., Pohl, K. (eds.) ICSE, pp. 582–591. IEEE/ACM (2013)
5. Cowie, J., Lehnert, W.: Information extraction. Commun. ACM **39**(1), 80–91 (1996)
6. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V., Aswani, N., Roberts, I., Gorrell, G., Funk, A., Roberts, A., Damljanovic, D., Heitz, T., Greenwood, M.A., Saggion, H., Petrak, J., Li, Y., Peters, W.: Text Processing with GATE (Version 6) (2011). ISBN: 978-0956599315. http://tinyurl.com/gatebook
7. Di Sorbo, A., Panichella, S., Alexandru, C.V., Shimagaki, J., Visaggio, C.A., Canfora, G., Gall, H.C.: What would users change in my app? summarizing app reviews for recommending software changes. In: Proceedings of the 2016 24th ACM SIGSOFT International Symposium FSE, pp. 499–510. ACM (2016)
8. Fang, X., Zhan, J.: Sentiment analysis using product review data. J. Big Data **2**(1), 1–14 (2015)
9. Feng, D., Shaw, E., Kim, J., Hovy, E.H.: An intelligent discussion-bot for answering student queries in threaded discussions. In: International Conference on Intelligent User Interfaces, pp. 171–177. ACM (2006)

10. Godfrey, J.J., Holliman, E.C., McDaniel, J.: SWITCHBOARD: telephone speech corpus for research and development. In: Acoustics, Speech, and Signal Processing, ICASSP-1992, vol. 1, pp. 517–520 (1992)
11. Guzman, E., Alkadhij, R., Seyff, N.: A needle in a haystack: what do Twitter users say about software? In: IEEE 24th International Conference in Requirements Engineering, pp. 96–105 (2016)
12. Guzman, E., Aly, O., Bruegge, B.: Retrieving diverse opinions from app reviews. In: 2015 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM), pp. 1–10, October 2015
13. Keertipati, S., Savarimuthu, B.T.R., Licorish, S.A.: Approaches for prioritizing feature improvements extracted from app reviews. In: Proceedings of the 20th International Conference EASE, pp. 33:1–33:6. ACM, New York (2016)
14. Kim, J., Chern, G., Feng, D., Shaw, E., Hovy, E.: Mining and assessing discussions on the web through speech act analysis. In: Proceedings of the Workshop on Web Content Mining with Human Language Technologies (2006)
15. Maalej, W., Nabil, H.: Bug report, feature request, or simply praise? On automatically classifying app reviews. In: 2015 IEEE 23rd International Requirements Engineering Conference (RE), pp. 116–125. IEEE (2015)
16. Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S.J., McClosky, D.: The Stanford CoreNLP natural language processing toolkit. In: Association for Computational Linguistics (ACL) System Demonstrations, pp. 55–60 (2014)
17. Martin, W., Sarro, F., Jia, Y., Zhang, Y., Harman, M.: A survey of app store analysis for software engineering. IEEE Trans. Softw. Eng., 1, 5555 (2016). doi:10.1109/TSE.2016.2630689
18. Morales-Ramirez, I., Perini, A.: Discovering speech acts in online discussions: a tool-supported method. In: Joint Proceedings of the CAiSE 2014 Forum, volume 1164 of CEUR Workshop Proceedings, pp. 137–144. CEUR-WS.org (2014)
19. Morales-Ramirez, I., Perini, A., Ceccato, M.: Towards supporting the analysis of online discussions in OSS communities: a speech-act based approach. In: Nurcan, S., Pimenidis, E. (eds.) CAiSE Forum 2014. LNBIP, vol. 204, pp. 215–232. Springer, Cham (2015). doi:10.1007/978-3-319-19270-3_14
20. Morales-Ramirez, I., Perini, A., Guizzardi, R.S.S.: An ontology of online user feedback in software engineering. Appl. Ontol. **10**(3–4), 297–330 (2015)
21. Neulinger, K., Hannemann, A., Klamma, R., Jarke, M.: A longitudinal study of community-oriented open source software development. In: Nurcan, S., Soffer, P., Bajec, M., Eder, J. (eds.) CAiSE 2016. LNCS, vol. 9694, pp. 509–523. Springer, Cham (2016). doi:10.1007/978-3-319-39696-5_31
22. Novielli, N., Strapparava, C.: Dialogue act classification exploiting lexical semantics. In: Conversational Agents and Natural Language Interaction: Techniques and Effective Practices, pp. 80–106. IGI Global (2011)
23. Panichella, S., Di Sorbo, A., Guzman, E., Visaggio, C.A., Canfora, G., Gall, H.C.: How can i improve my app? Classifying user reviews for software maintenance and evolution. In: IEEE International Conference on Software Maintenance and Evolution (ICSME), pp. 281–290. IEEE (2015)
24. Searle, J.R.: Speech Acts: An Essay in the Philosophy of Language, vol. 626. Cambridge University Press, Cambridge (1969)
25. Stolcke, A., Coccaro, N., Bates, R., Taylor, P., Van Ess-Dykema, C., Ries, K., Shriberg, E., Jurafsky, D., Martin, R., Meteer, M.: Dialogue act modeling for automatic tagging and recognition of conversational speech. Comput. Linguist. **26**(3), 339–373 (2000)

26. Villarroel, L., Bavota, G., Russo, B., Oliveto, R., Penta, M.D.: Release planning of mobile apps based on user reviews. In: Proceedings of the 38th International Conference on Software Engineering, pp. 14–24. ACM (2016)
27. Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesslén, A.: Experimentation in Software Engineering: An Introduction. Kluwer Academic Publishers, Norwell (2000)