Terms Extractions: An Approach for Requirements Reuse

Noor Hasrina Bakar, Zarinah M. Kasirun
Department of Software Engineering
University of Malaya
Malaysia
noorhasrina@siswa.um.edu.my,
zarinahmk@um.edu.my

Norsaremah Salleh
Kuliyyah of ICT
International Islamic University of Malaysia
Malaysia
norsaremah@iium.edu.my

***Abstract*—This paper presents a solution to a requirements reuse problem that utilises natural language processing and information retrieval technique. We proposed a semi-automated approach to extract the software features from online software review to assist the process to reuse natural language requirements. We have conducted an experiment to compare the manual feature extraction versus the semi-automated feature extraction. We used compilations of software review from the Internet as a source of this extraction process. The extracted software features are compared against the features obtained manually by human and the evaluation results obtained in terms of time, precision, recall, and F-Measure indicate a promising result.**

Keywords—features extraction, software reuse, requirements engineering

I. INTRODUCTION

Requirements for any existing system can be extracted and reuse for production of a similar new system. However, reuse of software features extracted from Software Requirement Specifications (SRS) is only viable to practitioners who have access to these software artefacts. SRS usually reside within company databases that are kept confidential and therefore makes it hard for external researchers to access and further explore its reuse potential.

Manual and ad hoc reuse of requirements can be very arduous, time consuming, labour intensive, and error-prone on the results. Additionally, when SRS are not available, valuable information from publicly available data such as software reviews from the Internet can be beneficial to requirements engineers. In this research, we propose a semi-automated process to extract software features from public data which can assist the process to reuse natural language requirements. Our aim is to compare the manual extraction process versus the proposed semi-automated process for extraction of software features. We adapted the approach from natural language processing and information retrieval in order to accomplish this task.

II. RELATED WORKS

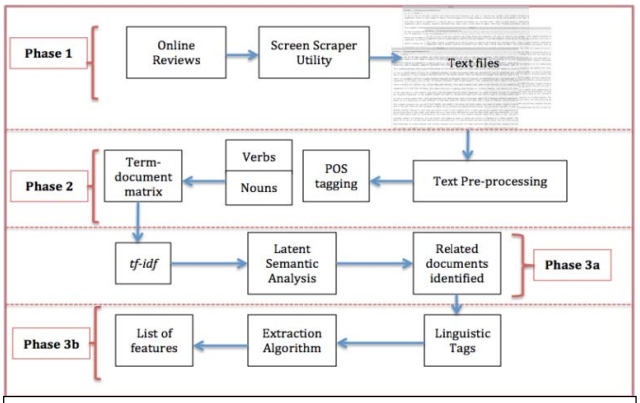
In related research, software features were extracted from various other forms of Natural Language Requirements (high-level requirements) when SRS are not accessible. For example, product brochures were used in [1], online product listings were used in [2], and the use of multiple web repositories were reported in [3]. In terms of extraction approaches, authors in [1] used contrastive analysis for the extraction of features that are related to components of software products. Hariri et al. in [4] have extracted the feature descriptors from online product listings and targeted at sentences for product descriptors, utilising association rules mining, and k-nearest neighbour to analyse the neighbourhood for a similar product in order to identify new features. Despite the rigorous explanation on the feature recommendation process, the work in [4] did not describe the extraction approach especially pertaining to handling the combination of parts of speech tags in sentences that can form features, which we think are very important for readers interested in this area to know.

Few works [5] [6] [7] in the requirements engineering area used the mobile app reviews to either extract the feature requests or for the purpose of re-designing the existing functionalities for the mobile apps for requirements evolutions. These related works that processed user reviews from mobile apps incorporated the combination of topic modelling such as Latent Dirichlet Allocation, LDA, and sentiment analysis to obtain knowledge about user opinion. Although various works exist in the area of feature extraction in requirements engineering, none of the works has reported the use of software reviews from the web as a source to initiate feature (high-level requirements) extractions in the context of requirements reuse. Compared to related work, our work focuses on extracting the features of the software products that can represent functionality of software being reviewed. (For more detailed related works, refer to our systematic literature review related to Feature Extraction Approaches that is available in [12]) .

978-1-4673-8611-1/15/$31.00 ©2015 IEEE

III. OUR APPROACH

We refer to the definition of features as a prominent or distinctive user visible characteristic of a software product [8]. In our research, we focus on extracting features or visible software characteristics that are related to the software functionalities, which can be extracted from the user reviews. Our approach is divided into three phases, as tabled out in Figure 1:



Phase 1: Assessing Requirements
Phase 2: Terms Extractions
Phase 3: Feature Identification:
Phase 3a: Identifying related documents
Phase 3b: Extraction of terms that represent software features

Figure 1: Feature Extraction Process

In Phase 1, by using open source screen scraper utility, we have scraped 32 compilations of software reviews pertaining to online learning posted by software experts at toptenreviews.com and superkids.com. Each of these reviews are not the first-hand user reviews like what was being used by [9], [7], or [6]. Their number of user reviews might appear bigger as compared to ours; however, the length of each reviews in the related works is shorter compared to the compilation of user reviews in our work. We did not extract the first-hand reviews from users because we want to filter out the user complaints and sentiments from the reviews. Most importantly, we want to focus on extracting information related to the software features available for reuse. Thus, using compiled reviews from experts will be a better choice to suit this purpose.

We used the documents being scraped in Phase 1 as the input to the terms extraction process. Fig. 2 lists out the step-by-step process used in Phase 2: Terms extraction. First, the scraped documents are pre-processed to remove stop-words, punctuations, special characters, and numbers. After that, we have used the Parts of Speech, POS Tagger in NLTK to tag the nouns and verbs from all reviews and compile their occurrences in a term-document-matrix, TDM. Verbs or nouns that occur too many times (occurrences of more than 10 times) are removed from the matrix because this obviously will not provide added weight to the matrix.

Step 1: Each document went through text pre-processing in Python to remove the stop-words, punctuations, numbers, and special characters.
Step 2: Apply the Part of Speech Tagging from NLTK[2] to the document and select the required terms (verbs and nouns).
Step 3: Remove selected terms that occur only once and twice.
Step 4: The remaining terms with its occurrences were tabulated in a term-document-matrix.

Figure 2. Phase 2: Term extraction process

The TDM is then passed to Phase 3: Feature Identification. Phase 3 comprises of two sub-phases: Phase 3a: Identifying related documents and Phase 3b: Extraction of terms that represent software features. With the TDM produced, we applied the term-frequency-inverse document frequency, *tf-idf* calculation. The *tf-idf* weight is a weight used in information retrieval and text mining to evaluate how important a word is to a document in a collection. The importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the collection [10]. By using the *tf-idf* values obtained, the position of related documents in document space can be obtained by applying the Singular Values Decomposition through the LSA implementation. We have described the detail for this process in our previous work [11]. As a result, unrelated documents were discarded: the documents that are clearly far from other documents in the document space (as computed by LSA) will not be taken into the next phase of the experiment. From this exercise, we confirmed that all 32 reviews actually came from four sub-categories (refer Table 1).

In Phase 3b, we used reviews from each software category and identified the terms that occur in the following linguistic tags: `<<adjective, noun>>` or `<<noun, adjective>>` AND `<<verb, adjective>>` or `<<adjective, verb>>` AND `<<verb, nouns, adjective>>`. Since we are interested to extract features that are mostly related to functionalities of the software, all possible forms of verb (past or present tense) combination with nouns and adjectives are considered in the linguistic tags. The output from Phase 3 will be a list of features extracted from the reviews and this list can be further clustered prior to feature modelling activity. Due to space limitation, the later parts of feature extraction process for requirements reuse (the clustering of features and the formation of feature model) as proposed in our previous work in [12] will not be presented in this paper.

IV. RESULTS AND DISCUSSION

To demonstrate the semi-automated feature extraction approach, we have conducted an experiment in the laboratory setting. The extraction algorithm was implemented by using Python 2.7 installed on Macintosh machine. To measure the accuracy of the proposed approach, we have engaged seven

teachers to read through the online software reviews and perform the manual extraction. We engaged teachers to conduct the manual feature extraction for two reasons. First, we wanted to know how long does it take to extract software features manually and we used teachers as the subject matter experts who can identify what a software can offer from reading the reviews. Second, we do not have any truth set of data and thus we engaged teachers from various disciplines that are related to the selected software categories to perform the manual extraction. What have been pulled out by the teachers are compiled and set as the truth data set. Furthermore, the manually extracted features are then compared to the ones extracted by the semi-automated approach.

In this section, we will firstly present the dataset used, followed by the comparison in terms of recall, precision, and time obtained from manual extraction versus the semi-automated approach.

A. Data Set

The following are demographics data being used in the experiment:

TABLE 1: DATASETS OF SOFTWARE REVIEWS

Learning Software Subcategory	# of software being reviewed	Length (# of sentences)	Word Lists *
Preschool Learning	10	426	1998
Algebra Learning	10	296	1144
Language and Reading Software	3	91	725
Creative Writing Software	9	440	1140
Total	**32**	**1253**	**5007**
** total number of distinct words after removing stop-words, numbers, punctuations, special characters, etc.*			

We chose online reviews for four categories of learning software, as illustrated in Table 1. Total word lists extracted after pre-processing are 5007 and the average length per review is about 39 sentences. Figure 3 illustrates the sample output on the features extracted after executing the semi-automated approach that matches with the one extracted manually.

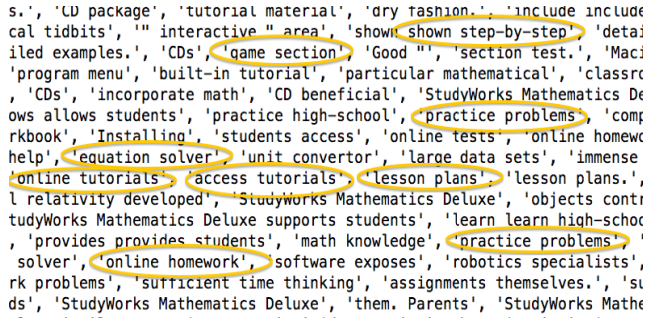


Figure 3. Sample output

B. Manual Extraction versus Semi-automated Approach

The manual extraction results obtained were compiled by the first author and the final list of features is used as truth data set. This truth data set is important for calculating the Recall, Precision, and F-Measure for the semi-automated approach. Table 2 details out the number of features extracted by teachers (manual) versus the number of features extracted by the semi-automated approach.

TABLE 2: NO. OF FEATURES AND TIME TAKEN FOR MANUAL VERSUS SEMI-AUTOMATED EXTRACTION

#	Online Learning Software Category	# of features extracted / estimate time taken (Phase 3b)	
		Manual	**Semi-Automated***
1	**Preschool Learning**	193 (2 hrs)	457 (9.8 secs)
2	**Algebra Learning**	182 (1hr 40 mins)	295 (6.1 secs)
3	**Language and Reading Software**	78 (55 minutes)	147 (5.5 secs)
4	**Creative Writing Software**	289 (1 hr 52 minutes)	534 (6.6 secs)
** we measured the time for performing the extraction only Phase 3b (that excludes the time for Phase 1 and 2)*			

Additionally, we also recorded the time taken for both approaches and obviously the semi-automated approach requires a shorter time to complete (see Table 2).

To measure the accuracy of the extracted features, we calculate Recall, Precision, and F-Measure. The following are the equation used to calculate Recall, Precision, and F-Measure:

$$Recall = \frac{No.of\ True\ Positives}{Sum\ of\ True\ Positive\ and\ False\ Negatives} \quad (1)$$

$$Precision = \frac{No.of\ True\ Positives}{Sum\ of\ True\ Positives\ and\ False\ Positives} \quad (2)$$

$$F - Measure = 2\ X\ \frac{Precision\ X\ Recall}{Precision + Recall} \quad (3)$$

It the context of feature extractions, we consider the following elements for the definition of the evaluation metrics: a) True Positive is defined as correctly returned features - what is extracted by the semi-automated process also exists in the truth data set. b) False Positive is defined as what is returned by the semi-automated process which does not exist in the truth dataset. c) False Negative is defined as actual features that are not returned by the semi-automated process.

Result for Recall, Precision and F-measure is tabulated in Table 3. It is worth noting that recall is held pretty high compared to the percentage for precision. We observe that on average the automated extraction approach extracted more than 70% of features that reside within the truth set.

Although the semi-automated approach did not suggest all possible features from the truth set, this automated approach can briefly provide some guidance for the requirements engineers to select the existing features for a production of a new software from a similar family.

TABLE 3: PRECISION, RECALL, AND F-MEASURE FOR AUTOMATED EXTRACTION RESULT

Online Learning Software Category	Recall	Precision	F-Measure
Preschool Learning	76.40%	33.60%	46.67%
Algebra Learning	70.30%	30.30%	42.35%
Language and Reading Software	79.50%	42.20%	55.13%
Creative Writing Software	80.30%	43.50%	56.43%

Additionally, requirements engineers may use the features extracted by the semi-automated tools as a starting point to kick off the production without having to start from zero on the requirements engineering process.

CONCLUSIONS AND FUTURE WORKS

In this paper, we have proposed a semi-automated approach that extracts features related to software functionality that may reside within user reviews on the Internet. We believe that this semi-automated approach can assist requirements engineers in the requirements engineering activities, especially in the context of requirements reuse. We have conducted a laboratory experiment that demonstrates the proposed approach and compared its performance with the manual extraction process. The evaluation conducted indicates a higher recall (more than 70%) but slightly lower precision. Although the semi-automated approach generated a promising recall observation, the lower precision value indicates that the semi-automated approach also generated some features that are not in the truth set, either possibly be missed by the teachers, or it has extracted some noises which can introduce some threats to validity. These results serve as a motivation for future improvements on the approach. One possible way to address this limitation is by applying the regular expression and collocation to the linguistic tags.

In the near future, the output from Phase 3 of the experiment will be used as an input to feature clustering process, where similar features will be clustered together and passed to the feature modelling activities.

ACKNOWLEDGEMENT

This research is funded by the Ministry of Higher Education Malaysia with Research Grant# FP050/2013A, with Assoc. Prof. Dr. Zarinah Kasirun as the principal investigator from the Department of Software Engineering, FSKTM of the University of Malaya.

REFERENCES

[1] A. Ferrari, G. O. Spagnolo, and F. Dell'Orletta, "Mining commonalities and variabilities from natural language documents," in *Proceedings of the 17th International Software Product Line Conference on - SPLC '13*, 2013, p. 116.

[2] J.-M. Davril, E. Delfosse, N. Hariri, M. Acher, J. Cleland-Huang, and P. Heymans, "Feature model extraction from large collections of informal product descriptions," *Proc. 2013 9th Jt. Meet. Found. Softw. Eng. - ESEC/FSE 2013*, p. 290, 2013.

[3] Y. Yu, H. Wang, G. Yin, and B. Liu, "Mining and recommending software features across multiple web repositories," in *Proceedings of the 5th Asia-Pacific Symposium on Internetware - Internetware '13*, 2013, pp. 1–9.

[4] H. Hariri, C. Castro-Herera, M. Mirarkholi, J. Cleland-Huang, and B. Mobasher, "Supporting Domain Analysis Through Mining and Recommending features from Online Product Listings," *IEEE Trans. Softw. Eng.*, vol. 39, no. 12, pp. 1736–1752, 2013.

[5] E. Guzman and W. Maalej, "RE 2014 - Appstore review and reuse," in *How Do Users Like This Feature? A Fine Grained Sentiment Analysis of App Reviews*, 2014, pp. 153– 162.

[6] L. V. G. Carreno and K. Windbladh, "Analysis of User Comments: An Approach for Software Requirements Evolution," in *International Conference of Software Engineering, ICSE 2013*, 2013, pp. 582–591.

[7] C. Iacob and R. Harrison, "Retrieving and Analyzing Mobile Apps Feature Requests from Online Reviews," in *Mining Software Repositories (MSR), 2013 10th IEEE Working Conference*, 2013, pp. 41–44.

[8] K. Kang, S. Cohen, J. Hess, W. Novak, and A. Peterson, "Feature Oriented Domain Analysis (FODA) Feasibility Study," Pittsburgh, PA, 1990.

[9] E. Guzman and W. Maalej, "How Do Users Like This Feature? A Fine Grained Sentiment Analysis of App Reviews," in *Requirement Engineering Conference 2014*, 2014, pp. 153–162.

[10] G. Salton and C. Buckley, "Term Weighting Approaches in Automatic Text Retrieval," *Inf. Process. Manag.*, vol. 24, no. 5, pp. 513 – 523, 1988.

[11] N. H. Bakar, Z. M. Kasirun, and H. A. Jalab, "Towards Requirements Reuse: Identifying Similar Requirements with Latent Semantic Analysis and Clustering Algorithms," in *Proc. of the Second Intl. Conf. on Advances In Computing, Communication and Information Technology- CCIT 2014.*, 2014, pp. 19–24.

[12] N. H. Bakar, Z. M. Kasirun, and N. Salleh, "Feature extraction approaches from natural language requirements for reuse in software product lines: A systematic literature review," *J. Syst. Softw.*, vol. 106, pp. 132–149, Aug. 2015.