Towards Automatic Requirements Elicitation from Feedback Comments: Extracting Requirements Topics Using LDA

Hitoshi Takahashi

Hiroyuki Nakagawa

Tatsuhiro Tsuchiya

Graduate School of Information Science and Technology Osaka University 1-5 Yamadaoka, Suita, 565-0871 Japan {t-hitosi, nakagawa, t-tutiya}@ist.osaka-u.ac.jp

Abstract

Feedback comments, such as mailing lists and reviews, contain beneficial suggestion for software developers. Recently, developers have received more and more feedback comments; but it is still difficult to extract beneficial comments from a large amount of e-mail message or reviews. Latent Dirichlet Allocation (LDA) is a promising way of topic modeling, which classifies documents according to implicit multiple topics. In this paper, we tried to apply a requirements elicitation based on LDA to two different sources, i.e., Apache Commons User List and App Store reviews, and discuss the feasibility of this approach. An interesting finding was that some usual stop words indicated requirements description. This suggests that these words should be removed from the stop word list before applying LDA.

1 Introduction

Feedback comments are beneficial resources for developers because these comments contain information about what they want. Recently, more and more feedback comments have been gathered due to the improvement of user's environment. As stated in [5], more than one million of reviews in Google Play are uploaded per a day.

A lot of feedback comments can be useful as references for development; however, we usually have to manually extract beneficial data such as implicit requirements from these resources. The size of the resources is often too large to deal with manually.

Automatic extraction of requirements from user's feedback comments that are described in natural languages would be desirable. Some applications of linguistic engineering technology to manage requirements in mass software development have received recent attention. Dag et al.

[7][10] introduced the Baan requirements management process, which finds the relationships between feedback comments and business requirements (objectives). Some studies addressed the extraction of requirements from feedback review comments of smartphone applications. Fu et al. [8] introduced a system that analyzes App Store reviews and identifies problems by topic modeling. The system also classifies feedback comments into individual functions. Guzman et al. [9] proposed an approach that introduces rating of each function from words and emotions associated with them. The latter two studies use Latent Dirichlet Allocation (LDA) [6] for topic modeling. These studies extract topics related to functions of the target application; however, to the best of our knowledge, there is no existing work for automatic requirements elicitation from the feedback comments.

As the first step of the automatic requirements elicitation from feedback comments, we introduce a requirements elicitation process from the feedback comments based on LDA topic modeling. We also apply this elicitation process to Apache Commons User List and App Store reviews, which are respectively of types e-mail messages and reviews, aiming to elicit topics related to requirements. The experimental results indicated that our approach still needs further improvement; however, the results also provided some finding about the requirements elicitation from the feedback comments.

This paper is organized as follows: Section 2 describes the research questions in this work. Section 3 explains our requirements elicitation process from the feedback comments. Section 4 presents the results of experimental elicitation from two different types of feedback comments. Section 5 discusses the feasibility of our approach, and Section 6 concludes the paper.

Table 1. Target resources.							
Resource	E-mail	Review					
Language	English	English					
Main subject	technical questions,	evaluation (rating),					
	announcement	bug report					
Size	relatively large	usually small					
Frequency of	few	many					
requirements							

Table 1. Target resources

2 Research Questions

Our goal is to establish automatic requirements elicitation from feedback comments. In particular, we address the following two research questions.

- **RQ1:** Can we classify feedback comments into those that include requirements and those that do not include?
- **RQ2:** Does the quality of extracted requirements vary depending on the types of the feedback comments?

We use LDA for the requirements elicitation from the feedback comments. LDA constructs topics from the resource documents, i.e., feedback comments in this study. These topics indicate implicit characteristics of the documents and enable to classify documents. Therefore, RQ1 corresponds to the question whether we can extract topics related to requirements description.

As for RQ2, feedback comments can be roughly classified into e-mail messages and reviews. Table 1 shows their characteristics. In this study, we try to clarify whether the performance of the LDA classification depends on the resource type, i.e., e-mail messages and reviews.

3 Elicitation Process

Figure 1 illustrates our elicitation process. This elicitation consists of four activities, *lemmatizing, topic modeling using LDA, requirements topic elicitation,* and *requirements comments extraction.* The following sections explain these activities.

3.1 Preparation: Lemmatizing

Feedback comments that we deal with in this study are written in English and contain verb words that are conjugated. For example, "wishes" is the third person form of "wish" and they have same meaning but LDA recognize them as completely different words. We lemmatize conjugated with the WordNet Lemmatizer [11][4], which can be used to remove affix from the input word.

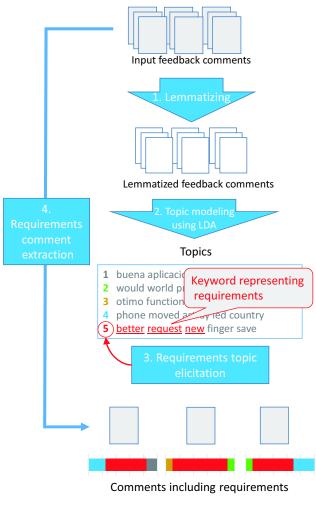


Figure 1. Elicitation process.

3.2 Topic Modeling Using LDA

We use Latent Dirichlet Allocation (LDA) [6][12] for topic classification. LDA is a topic model and widely used for unsupervised word classification. In LDA, topic distribution generates the topic for a word, and the topic for a word generates the specific word. Figure 2 presents the graphical model of LDA. The symbols in the figure correspond to the following concepts:

- ϕ : word distribution for topic
- θ : topic distribution for document
- z: topic for word

w: word

 α , β : hyper parameter. These parameters are given or usually estimated by a machine learning tool.

K: the number of topics

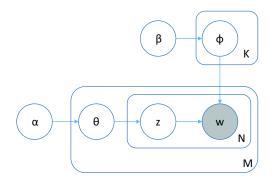


Figure 2. Graphical model representation of LDA.

- M: the number of documents
- N: the number of words in mth document

Hyper parameter α determines the topic distribution for document θ , and the topic z is determined according to θ . Another hyper parameter β determines the word distribution for topic ϕ , and finally the word w is determined according to z and ϕ .

In order to construct the topic model that can classify the feedback comments, we use LDA in this study according to the following steps:

- Step 0. (Preparation) Give a set of documents (M and N are determined) and set the number of topics K.
- Step 1. Set the default topic for each word in all comments.
- Step 2. Select each word w from the comments.
- **Step 3.** Change the topic z for the word w according to the probability P shown in Eq. (1).
- **Step 4.** Repeat 2. and 3. until N_t^- and N_{mt}^- in Eq. (1) are converged.
- **Step 5.** Output ϕ as the word distribution for topic and θ as the topic distribution for comment.

$$P(z=t|Z^-, W, \alpha, \beta) \propto \frac{\beta + N_{tw}^-}{\beta V + N_t^-} (\alpha_k + N_{mt}^-)$$
(1)

 Z^- : set of topics of all words excluding the word w.

W: set of all words in all documents.

 N_t^- : the number of words in all documents whose topics are t.

 N_{tw}^- : the number of word w in all documents whose topic is t.

 N_{mt}^{-} : the number of words in selected document m whose topics are t.

 α_k : the k th (topic k's) parameter α .

V: the number of words in all documents.

We use MALLET [3], a tool package for the topic classification based on LDA. This tool also has the word tokenization and unnecessary word removal functions. We input feedback comments to MALLET, and MALLET outputs ϕ , θ , and z by constructing the topic model based on LDA.

3.3 Requirements Topic Elicitation

After the topic model is constructed by MALLET, we find the topics related to the requirements by using ϕ . We identify the words likely to be contained in the requirement description and find topics that contain many words related to the requirements description as the topics related to the requirements.

3.4 Requirements comment extraction

We can extract possible comments that include requirements description based on the topics related to the requirements acquired in the previous activity. We use θ and find the comments that has a high relationship to the topic related to the requirements.

4 Experiment

4.1 Elicitation from Different Resources

We applied our elicitation process to two different sources, i.e., Apache Commons User List and App Store reviews. Apache Commons User List [1] is a mailing list for contacting users and developers, supported by Apache Software Foundation. Most of the messages describes technical questions and answers about software belonging to apache commons. Other mails are for the announcement of the latest version and questions about software functions; therefore, there are few messages related to requirements for new functions. Figure 3 illustrates an example e-mail message that contains the requirement for adding new output option to the CSVPrinter class.

App Store and Google Play are well-known review platforms. In this experiment, we use reviews in App Store [2] as a resource of reviews. The reviews are composed of five parts: *title, rating, author, date*, and *body text*.

The review shown in Figure 4 is a review for Google Maps. This review requests an additional function to rename designated places to Google Maps.

Subject: [CSV] Wish: format-specific date generation

Hi -

It would be useful if printing a Java Date or Calendar to a CSVFormat. EXCEL CSVPrinter would generate output that Excel recognises as a date-and-time. For example the following

PrintWriter outputWriter = new PrintWriter(new File-OutputStream("output.csv"));

which Excel only treats as a string. (It will recognise e.g. yyyy/mm/dd as a date but I wouldn't know where to look for a definitive set of formats it will consume.) Ditto probably printing Calendar.getInstance(), or the new Java 8 LocalDate etc. classes.

One argument against though is then the library perhaps ought to do the reverse, i.e. spot that it has been passed a date in and construct a Date class for the value at parse time which may be expensive and often unnecessary.

. . .

Figure 3. An e-mail message containing requirements.

We collected the same number of messages from Apache Commons User List and Google Maps reviews in App Store. We applied our elicitation process and extracted topics and comments related to the requirements. We, in particular, collected 300 messages and reviews from Apache Commons User List and Google Maps reviews in App Store, respectively. Among them, 17 messages contain requirements ($\#R_{max_E}$) and 47 reviews contain requirements ($\#R_{max_R}$).

We constructed topic models under the conditions that the topic size is 20, 40, and 100, respectively, where the topic size determines how many topics LDA generates. We regarded an extracted topic (T) as a topic related to requirements if it contained at least two words related to requirements. By using the extracted topics T, we extracted comments whose topic distribution for one of the extracted topics related to requirements are over 0.1 as the comments related to requirements (C).

We made an optimization that made use of the characteristics of the words related to requirements. Since we believe that the stop words of MALLET, which are the most common words and filtered out before topic model construction, include some words, e.g., *a, able, about, above*, and so on, - Title : Great App ! -

This app is incredible ! But it could be better if i could rename my places

Figure 4. A review containing requirements.

able, appropriate, appreciate, asking, ask, awfully, because, better, best, cannot, can, contains, containing, contain, considering, consider, currently, could, different, enough, except, help, hopefully, if, like, need, needs, necessary, new, normally, please, shall, should, toward, towards, tries, trying, try, unfortunately, useful, want, wants, will, why, would

Figure 5. The words removed from stop word list.

related to the requirements, we excluded some words from the stop word list. Figure 5 represents the words that we excluded from the stop word list.

4.2 Experimental results

Table 2 lists the results of applying our elicitation process, where #*Requirement topics* is the number of the extracted topics related to requirements (#*T*), #*Extracted comments* is the number of comments that are extracted by our elicitation process (#*C*), and #*Requirement comments* is the number of the extracted comments that were indeed related to requirements among C (#*R*).

Table 2 demonstrates that the elicitation from reviews worked more precisely than that from e-mail messages. This table also indicates that the topic size affects the precision and recall of the requirements elicitation. In this experiment, 40 topics was the most suitable size for constructing the topic model. Another finding was that the stop word list modification made improvements of precision and recall in many cases.

Figures 6 and 7 illustrate the extracted topics related to requirements in the cases of Apache Commons User List and App Store reviews with 40 topics extraction and stop word list modification, respectively. As shown in both the figures, we extracted four topics related to requirements, respectively. We regarded some words excluded from the stop word list as the words related to requirements. For example,

Table 2. Requirements elicitation results. Here, Precision = #R / #C, $Recall = \#R / \#R_{max_E}$ (for *E-mail*) **or** $Recall = \#R / \#R_{max_R}$ (for *Reviews*), and *F-measure* = 2 · *Precision* · *Recall* / (*Precision+Recall*).

Resource	Topic size	Stop word	#Requirement topics (#T)	#Extracted comments (#C)	#Requirement comments (#R)	Precision	Recall	F-measure
E-mail	20	default	1	22	0	0.000	0.000	0.000
		modified	3	81	2	0.0247	0.118	0.041
	40	default	1	23	1	0.0435	0.059	0.050
		modified	4	107	7	0.0654	0.412	0.113
	100	default	2	22	0	0.000	0.000	0.000
		modified	8	96	3	0.0313	0.176	0.053
Reviews	20	default	0	0	0	0.000	0.000	0.000
		modified	5	290	41	0.141	0.976	0.247
	40	default	1	23	12	0.522	0.286	0.369
		modified	4	111	29	0.261	0.690	0.379
	100	default	0	0	0	0.000	0.000	0.000
		modified	3	30	7	0.233	0.167	0.194

Topics related to requirements

3 {evaluation integrator number problem univariate integrator **could** fitting **need would** math **if should** case integration rule doe class gaussintegrator default}

12 {**can** scxml log **if** time scripting improvement **need** common method class **could will** default implementation simplecontext agent issue static}

22 {ascert file digester size inumdestinations node div listnodepathdata xml version rob www set **if** add **need** coefficient default true}

35 {org commons apache user mail unsubscribe help additional command wrote **can if** doe gmail **would should** issue http pm }

Figure 6. Extracted topics from Apache Commons User List (topic size = 40, using the modified stop word list).

since the word "could" may express the hope, and the word "if" is used for expressing subjunctive mood, we defined both words as the words related to requirements.

5 Discussion

We will now answer to the research questions based on the experimental results, and then discuss the limitations.

As for RQ1: "Can we classify feedback comments into those that include requirements and those that do not in- Topics related to requirements

10 {**need why** annoying log number phone info stop ad pop business data travel broke rating chase profit stay company}

12 {ca amazing user **will** saved friendly **if** running star review le rename **could** point accurate **why** real working live}

19 {car real information live doe wonderful happy navigational available waze constantly thousand supposed cart traveler **want because** point exceptional}

33 {street view **feature** version address close found exit **able** bad **new** streetview switch number interface ruined wo half level}

Figure 7. Extracted topics from App Store reviews (topic size = 40, using the modified stop word list).

clude?", the usefulness of the topic modeling approach depends on whether we can extract highly precise topics related to requirements. We extracted the topics that seems to be related to requirements from both e-mail messages and reviews in some cases in our experiment; however, the adequate topic extraction seemed to require certain constraints. As the findings from our experiments, the determination of topic size and the stop word list modification should be taken into account. The topic size affects the accuracy of the classification of comments. Small topic size squeezes

multiple topics into a topic; large topic size, on the other hand, constructs exceedingly decomposed topics.

We should also consider the characteristics of the words related to requirements. Words in the general topics, such as features and domains, are usually nouns and verbs. These topics are relatively easy to be extracted by using the topic modeling techniques including LDA. However, when we consider the topic related to requirements description, we have to deal with not only nouns and verbs but also auxiliary verbs, such as can, will, could, and would. Moreover, we believe that we should extract some of multiple-topicconcerned nouns and verbs, such as "need" and "like", as the words in the requirements topic. Most of these words are included in the stop word list, which is defined for elaborating topics to be extracted. Therefore, we believe that we should exclude words related to requirements from the stop word list; however, such exclusion may also inject the ambiguity of topics to be extracted.

As for RQ2: "Does the quality of extracted requirements vary depending on the types of the feedback comments?", as roughly summarized, the elicitation from reviews tends to score higher recall rate than the extraction from e-mail messages; however, it also tends to extract excessive quantities of comments. A possible reason is the contents size. Since review comments are generally short, the extracted topics from reviews tend to cover large amount of comments. Further precise topic classification should be required for improving the precision rate.

The experiment results demonstrate that eliciting requirements comments from e-mail messages is relatively difficult for our current process. Comparing Figures 6 and 7, the extracted topics from e-mail include words related to more decomposed features. A possible reason is the diversity of the contents. Comments in e-mail, such as Apache Commons User List in our experiment, generally contains more words than reviews, to explain more detailed situation, sometimes attaching source code, which includes the method and class name to the e-mail messages. This situation may hamper the construction of topics.

6 Conclusions

As the first step of the automatic requirements elicitation from feedback comments, we defined a requirements elicitation process from the feedback comments based on LDA topic modeling. We applied the elicitation process to two different sources, i.e., e-mail messages and reviews, and discussed the feasibility of our approach. The experimental results suggest that our current elicitation process has some limitations but also has a possibility of providing an automatic mechanism of requirements elicitation from review comments. We also found that effective requirements elicitation requires the stop word list modification. One of our primary on-going studies is further improvement of precision and recall value of our elicitation. We plan to discuss the algorithms for the adequate topics extraction mechanism. A comment extraction mechanism from the acquired topics should be refined. We will also conduct case studies in the large amount of feedback comments to discover further findings for the elicitation. We believe that automatic processes for dealing with huge feedback such as our approach help us adapt to recent continuous software delivery.

References

- [1] Apache commons mailing lists. http://commons.apache.org/mail-lists.html.
- [2] App store. https://itunes.apple.com/us/genre/ios/id36?mt=8.[3] MALLET homepage. http://mallet.cs.umass.
- edu/. [4] WordNet A lexical database for English. hhttp://
- wordnet.princeton.edu/.
- [5] AppTornado GmbH. Number of available android applications. http://www.appbrain.com/stats/ number-of-android-apps/.
- [6] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. J. Mach. Learn. Res., 3:993–1022, Mar. 2003.
- [7] J. N. o. Dag, V. Gervasi, S. Brinkkemper, and B. Regnell. A linguistic-engineering approach to large-scale requirements management. *IEEE Softw.*, 22(1):32–39, Jan. 2005.
- [8] B. Fu, J. Lin, L. Li, C. Faloutsos, J. Hong, and N. Sadeh. Why people hate your app: Making sense of user feedback in a mobile app store. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 1276–1284, New York, NY, USA, 2013. ACM.
- [9] E. Guzman and W. Maalej. How do users like this feature? a fine grained sentiment analysis of app reviews. In 2014 IEEE 22nd International Requirements Engineering Conference (RE), pages 153–162, Aug 2014.
- [10] J. Natt och Dag, V. Gervasi, S. Brinkkemper, and B. Regnell. Speeding up requirements management in a product software company: linking customer wishes to product requirements through linguistic engineering. In *Requirements Engineering Conference, 2004. Proceedings. 12th IEEE International*, pages 283–294, Sept 2004.
- [11] J. Perkins. Python Text Processing with NLTK 2.0 Cookbook. Packt Publishing, 2010.
- [12] I. Porteous, D. Newman, A. Ihler, A. Asuncion, P. Smyth, and M. Welling. Fast collapsed gibbs sampling for latent dirichlet allocation. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 569–577, New York, NY, USA, 2008. ACM.
- [13] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning. Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, EMNLP '09, pages 248–256. ACL, 2009.