Improvement of User Review Classification Using Keyword Expansion

Kazuyuki Higashi
Graduate School of Information Science and Technology
Osaka University
Osaka, Japan
Email: k-higasi@ist.osaka-u.ac.jp

Hiroyuki Nakagawa
Graduate School of Information Science and Technology
Osaka University
Osaka, Japan
Email: nakagawa@ist.osaka-u.ac.jp

Tatsuhiro Tsuchiya
Graduate School of Information Science and Technology
Osaka University
Osaka, Japan
Email: t-tutiya@ist.osaka-u.ac.jp

***Abstract*—Application users can submit reviews for downloaded applications. Recently, developers have received more and more user reviews. However, it is still difficult to extract beneficial comments from a large amount of reviews. Latent Dirichlet Allocation (LDA) is a promising way of topic modeling, which classifies documents according to implicit multiple topics. However, there is a gap between the documents that the developer wants to extract and the document extracted by LDA. In this paper, we propose a method to extract documents of each category, such as requirements descriptions or bug reports, more accurately. Our method first decomposes the topics. Then, the method uses the keyword list which is a set of semantically similar words collected by word2vec, to integrate the decomposed topics. We apply our method to the applications user reviews in Apple Store and demonstrate the validity of it. Our approach can help application developers to extract beneficial information.**

I. Introduction

With popularity of smartphones and tablets, mobile application stores, such as Apple Store, Google Play Store, and Windows Phone Store, have been growing. Over two million applications on the Apple Store exists, and those applications have been downloaded over 130 billion times [1].

These application stores allow users to submit reviews for downloaded applications in form of star ratings and text reviews. User reviews for downloaded applications are beneficial resources for developers because these reviews contain information, such as user requirements, bug reports, and evaluations of specific features. However, the amount of reviews is too large to extract beneficial information manually. Some popular applications receive hundreds of user reviews every day. From this background, it is needed to analyze user reviews more efficiently. In order to extract useful information from documents written in natural language, Latent Dirichlet Allocation (LDA) for topic modeling can be used. LDA is one of the most famous topic modeling techniques, which can classify the set of documents according to implicit multiple topics. When applying LDA to user reviews, LDA is good at classifying topics related to functions, such as stability and design. However, it is difficult to correctly gather requirements descriptions or bug reports in the same topic because they are not functions. These reviews are often distributed into multiple topics. Therefore, we try to extract such crosscutting topics.

In this paper, we propose a method that allows to more accurately extract documents of crosscutting topics that developers need. Our approach is improving LDA by decomposing the topics and combining semantically similar topics. We apply this process to Facebook for iOS user reviews and compare it with the general LDA. The experimental results indicate that our approach can extract the documents more accurately than the general LDA.

This paper is organized as follows: Section II discusses related work. Section III gives the overview of our approach. Section IV gives the background of this study by providing the explanation of Latent Dirichlet Allocation (LDA). Section V explains our method of user reviews classification and keyword expansion. Section VI presents the results of experimental extraction from user reviews. Section VII discusses the feasibility of our approach, and Section VIII concludes the paper.

II. Related work

Several studies analyzed reviews in application stores to give findings to application developers. Iacob et al. [2] evaluated user reviews about change requests and discovered that 23% of user reviews describe feature requests. Chen et al. [3] devised AR-MINER which is an approach to filtering and ranking informative reviews, and demonstrated that, on average, 35% of reviews contain informative content. These papers motivated our work because a noticeable percentage of user reviews contain useful information for developers.

Guzman et al. [4] proposed an approach that introduces rating of each function from words and user's sentiments by associating each other. Fu et al. [5] introduced a system that analyzes application reviews and identifies problems such as stability issues or cost by topic modeling. Palomba et al. [6] classified user reviews and grouped user reviews linking source code components. While these studies use Latent Dirichlet Allocation (LDA) for topic modeling and extract topics related to functions of the target application, our work, to the best of our knowledge, aims to improve LDA to accurately

DOI reference number: 10.18293/SEKE2018-047

classify documents according to crosscutting topics, such as requirements descriptions and bug reports.

There are some studies that combine LDA with word embedding, which is a set of feature learning techniques in Natural Language Process. Moody [7] devised lda2vec which is a model that learns word vectors jointly with Dirichlet-distributed latent document-level mixtures of topic vectors. Li et al. [8] devised TopicVec which is a generative model combining LDA and word embedding, with the aim of exploiting the word collocation patterns both at the level of the local context and the global document. For sets of short sentences such as user reviews, feature learning is difficult. Therefore, these models are not suitable for the purpose of this study.

III. Background

Latent Dirichlet Allocation (LDA) [9][10] is a topic model, which can classify documents written in natural language. LDA can be applied to various subjects such as news articles, feedback comments, and microblogging. In LDA, we assume that words are generated by topics and that those topics are mixed within a document. Figure 1 represents the graphical model of LDA. The variables in the figure correspond to the following concepts:

α: hyper parameter about topic distribution
θ: topic distribution for document
z: topic for word
β: hyper parameter about word distribution
ϕ: word distribution for topic
w: word
K: the number of topics
M: the number of documents
N: the number of words in m-th 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. Finally, the word w is determined according to z and ϕ.

In order to construct the topic model that can classify the user reviews, LDA is used 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 a default topic for each word in all documents.
- **Step 2:** Select each word w from the documents.
- **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 a document.

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

where

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

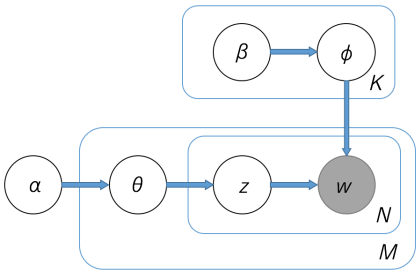

Fig. 1. Graphical model representation of LDA.

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.

Inputting the set of documents, LDA can output ϕ, θ, and z by constructing the topic model.

IV. Approach

Figure 2 shows the schematic view of our approach using an example. Each circle represents a topic. The figure shows the classification of topics with the general LDA and our method. We try to extract topics related to *bug* by the general LDA. However, if we extract one topic, there are documents related to *bug* that can not be extracted. If we extract two topics, we extract the documents unrelated to *bug*. Therefore, it is necessary to decompose the topics. By classifying topics finely as shown and extracting topics in consideration of the meaning of words, documents related to *bug* can be extracted more accurately. In order to obtain the meaning of words, we use word2vec.

Word2vec [11][12] is an unsupervised learning algorithm for learning distributed representations of words in a vector space using a neural network model. Each word in the document can be learned from surrounding words. Distributed representations of words help to improve performance in natural language processing tasks. For example, spatial distance between words describes the similarity between the words semantically and syntactically.

Training learns representations for each word w_t (the t-th word in a corpus of size T) so as to maximize the average log likelihood Eq.(2).

$$\frac{1}{T}\sum_{t=1}^{T} \log p(w_t \mid w_{t-c}^{t+c}) \quad (2)$$

c is the size of the training context (window size). w_{t-c}^{t+c} is the set of words in the window of size c centered at w_t. Continuous bag-of-words (CBOW) architecture predicts the current word based on the context. CBOW defines w_{t-c}^{t+c} as Eq.(3).

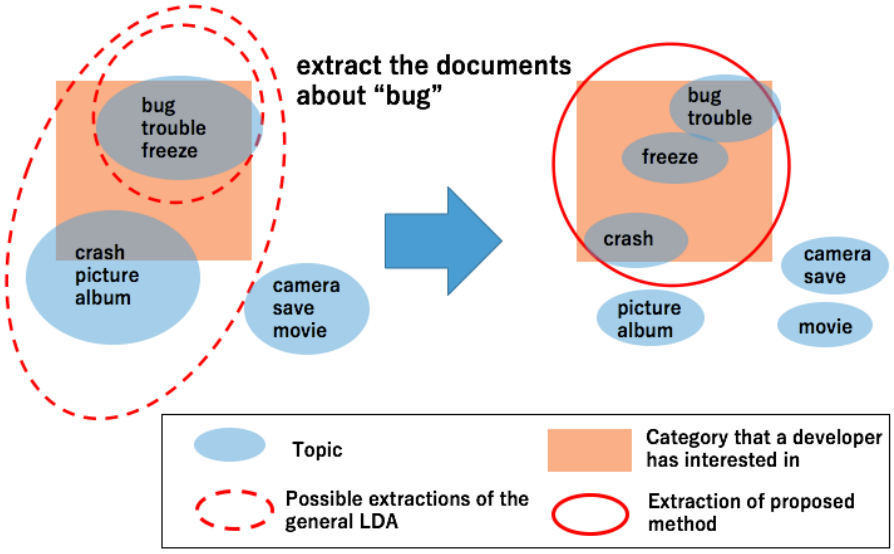

Fig. 2. Our approach is based on the decomposition of topics and binding decomposed topics using keyword expansion

$$p(w_t \mid w_{t-c}^{t+c}) = \frac{exp({e'_{w_t}}^{\mathrm{T}} \cdot \sum_{-c \le j \le c, j \neq 0} e_{w+j})}{\sum_w exp({e'_w}^{\mathrm{T}} \cdot \sum_{-c \le j \le c, j \neq 0} e_{w+j})} \quad (3)$$

where e_w and e'_w are the input and output vector representations of w.

V. Review Classification Using Keyword Expansion

In this section, we explain our method to classify reviews more accurately by using topic modeling. First, we collect the user reviews for a specific application and extract the title and text body from each review. Then, we preprocess the text data to remove the noise for topic modeling. Afterwards, we classify user reviews based on LDA.

We, in particular, classify the reviews more finely than in the general LDA. We can fragment reviews by setting larger value to the parameter *topic size* of LDA. In order to obtain the meaning of words, we collect semantically similar words by word2vec. We call this process keyword expansion. Finally, we select topics for each category we want to extract by using keyword list which is a set of semantically similar words.

Algorithm 1 explains the details of our method. We input the topic size, i.e., the number of topics to be generated, and keywords to the algorithm. The algorithm outputs documents belonging to the categories that the given keywords specify.

A. Topic Modeling Using LDA

We use Latent Dirichlet Allocation (LDA) to classify user reviews. LDA is one of the most widely used topic models, to classify user reviews.

We use MALLET [13], a tool package for the topic modeling based on LDA. This tool also has the word tokenization and unnecessary word removal functions. We input user reviews to MALLET, and MALLET outputs following results by constructing the topic model based on LDA:

Algorithm 1 Review Classification Using Keyword Expansion

```
input the number of topics larger than the general LDA
topic classification
for n in prepared keyword list do
  for top words in topics do
    calculate similarity between each keyword and top word
  end for
  sum similarity of each word
end for
add the top words of similarity to the keyword list
for topics do
  calculate the number of occurrences of the keyword list
  O = sum the number of occurrences
  if O ≥ threshold then
    output the documents in the topic
  end if
end for
```

- topic distribution for each review
- topic for each word
- word distribution for each topic (top words for each topic)

As shown in Figure 2, we intentionally obtain further segmentalized topics than those of the general LDA. For this purpose, we use a larger topic size than the size that we use in the general LDA modeling. As a result, we can obtain a number of smaller size topics than those of the general LDA modeling.

B. Keyword Expansion

After the review classification, our method collects topics that meet with categories, which a developer is interested in. First, the developer prepares keywords that represent categories. Our method finds similar words to the given keywords.

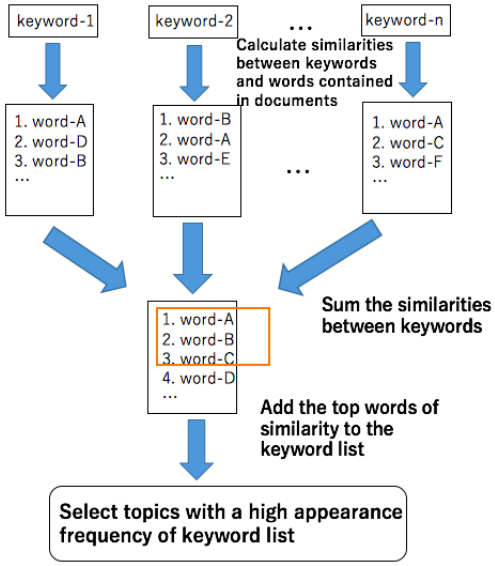

Fig. 3. Keyword expansion process

To obtain such similar words, our method uses word2vec, which can generate word vectors. By using word2vec, it is possible to calculate similarity between words and acquired similar words. Training data is full text of English Wikipedia [14] because the source covers various fields. After learning word vectors, when we input two words, the similarity between the words can be calculated. We use gensim [15], which is a package for python to learn word2vec model.

Figure 3 shows our keyword expansion process. When we find topics that belong to each category, we use keyword list obtained by keyword expansion. LDA can output topics for words and the number of occurrences of each word in each topic. We calculate the number of occurrences in each topic of the keyword list respectively. Then, the number of occurrences of each keyword is summed up for each topic, and topics whose sum is equal to or larger than the threshold are selected. Finally, we extract documents belonging to the selected topics.

VI. Experiment

We compared our method with the general LDA to evaluate it. We applied our method to Facebook user reviews, and extracted documents of three categories: *requirements*, *bug*, *resource*. *Requirements* and *bug* are crosscutting categories, and *resource* is one of the general functions. *Requirements* type review is a review that requests changes or improvement of specific features, and desires to add a new function. *Bug* type review is a review about application problems, stability issues, bugs of specific function. *Resource* type review is a review about the effect of application use on hardware, such as battery consumption and memory usage.

A. Dataset

We applied our approach to application user reviews to evaluate it. Apple Store is a well-known review platform. In this experiment, we used 1200 user reviews of Facebook for iOS in Apple Store from August 5 to September 8, 2015. The reviews in Apple Store are composed of five parts: title, rating, author, date, and body text. We extracted titles and body texts from the reviews. The reviews shown in Figure 4 are reviews for Facebook. The first one is about *requirements* that the user wants to stop turning off music. The second one is about *bug* that freezes the device when using the application. The third one is about *resource*, claiming that the application drains battery with background activity. Among 1200 reviews, 270 reviews should be classified into the *requirements* category, 447 reviews should be classified into the *bug* category, 40 reviews should be classified into the *resource* category.

Title: Works, but need to stop turning off Music
Body: While listening to music using the default music app on iOS 9.0.1 opening Facebook will pause the music no matter if the music is playing over the Built in Speaker, Headphones, or over Air-Play to an Apple TV. Please Fix. Very annoying.

Title: Causing my iPhone to freeze
Body: App freeze When watching videos or scrolling down though my feeds. While frozen my iPhone power button won't work and once the time out causes ur screen to go to sleep mode, u won't be able to turn it on till about 3-4 mins. Anyone having this issue?

Title: Battery killer
Body: This app is destroying my battery. Even with background refresh off and location off it still plows through battery with background activity.

Fig. 4. User reviews of each category

B. Preparation

In order to classify documents more accurately, we pre-processed the documents. English documents contain very common words (e.g., "a", "for", "is", and "that"), which are noisy to NLP activities. We removed these words as stop words from the documents. We used stop word list of MALLET, and modified it. Takahashi et al. [16] demonstrate that it is possible to make topics related to requirements likely to appear by removing several words related to the requirements from stop word list. Figure 5 represents the words that we excluded from the stop word list.

C. User reviews classification process

First, we used the general LDA as the baseline method. In this method, we constructed topic models under the conditions that the number of topic is 20. We prepared keywords which

TABLE I
KEYWORD EXPANSION RESULTS : TOP SIMILAR WORDS AND SIMILARITY FOR EACH CATEGORY

requirements		bug		resource	
want	1.41598253025	drop	0.910286822769	devices	1.07031276398
wish	1.34129323342	fix	0.880712643248	functionality	1.0634093351
hopefully	1.10542239509	glitches	0.826809114425	storage	1.0331266395
try	0.965955479289	overload	0.8121774242	cache	0.965408862186
help	0.944442417494	kill	0.76876851121	user	0.942083889095
sigh	0.920252507561	reset	0.765789424058	load	0.892174335253
feel	0.902822493972	trouble	0.763711373632	function	0.872954556545
will	0.883233011125	stuck	0.722607486066	software	0.836354576718

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

Fig. 5. The words removed from stop word list.

are related to category. Topics are selected by keywords. The following keywords are prepared for this experiment.

- **requirements :** please, need, hope
- **bug :** bug, crash, freeze
- **resource :** battery, memory, data

We selected topics if these keywords are in the top words of each topic.

In our method, we applied LDA under the conditions that the number of topic is 40 to break down the topic more. Then, we expanded keywords by using word2vec of gensim.

We constructed word2vec model under the conditions that learning model is CBOW, the dimensions of the vectors is 400, the size of window is 5, and other conditions are default of gensim.

We calculated similarity between each keyword mentioned above and the top 20 words of each topic. The similarities of the three keywords are summed, and the top three words are added as new keywords. Table I lists the results of keyword expansion. After applying word2vec to the user reviews, we acquired the top similar words and the similarity between these words and the prepared keywords for each category. We acquired the following words by keyword expansion.

- **requirements :** want, wish, hopefully
- **bug :** drop, fix, glitches
- **resource :** devices, functionality, storage

Topics for each category are selected based on the number of appearances in each topic of the keywords in the keyword list obtained by the keyword expansion. Total number of appearances in each topic of all keywords is summed up and top topics are selected. If the total number of appearances was less than 30, the topic was not selected. The number 30 was decided based on preliminary experiments.

D. Experimental results

Table II lists the results of applying our method. In order to evaluate the effectiveness of the proposed method, we check whether the extracted documents of each category actually correspond to the content of that category. We investigated the total number of documents in each category manually. After applying our approach, we counted the number of documents of extracted topics and the number of documents belonging to each category, out of the documents of the extracted topics. In order to evaluate our method, we used precision, recall, and F-measure as metrics. Precision is the rate of retrieved documents that are relevant to the category, and recall is the rate of the relevant documents that are successfully retrieved. F-measure is the harmonic average of the precision and recall.

Table II demonstrates that our method improves F-measure for all categories compared with the general LDA, especially *requirements* category.

VII. DISCUSSION

In this section, we discuss our results and describe the limitations and threats to validity.

Our method improves F-measure for all categories compared with the general LDA, especially *requirements* category. User reviews representing *requirements* have many feature words, such as, ”please”, ”want”, ”hope”, ”wish”, and ”need”, and sometimes there is no specific word, such as, “Add the new button to save a picture!”. Therefore, it is difficult to gather these reviews on the same topic. In the general LDA, unrelated documents are often mixed in the topic representing the *requirements*, resulting in a problem that precision is low. However, in our method, by preparing many keywords, it is possible to acquire the reviews of *requirements* more accurately. Our method also improves the F-measure of *bug* category. Since feature words representing *bug* are limited, precision and recall are higher than *requirements*. Our method improves the F-measure of *requirements* category slightly compared with the other two categories. There are few words representing *resource*, such as battery and memory, and documents tend to gather on the same topic.

Based on these results, our method is effective to extract crosscutting categories, such as *requirements* and *bug*. On the

TABLE II

DOCUMENT EXTRACTION RESULTS : #TOTAL NUMBER OF DOCUMENTS IN EACH CATEGORY ($\#D$), #THE NUMBER OF EXTRACTED TOPICS($\#T$), #THE NUMBER OF DOCUMENTS OF EXTRACTED TOPICS ($\#D_t$), #THE NUMBER OF DOCUMENTS BELONGING TO EACH CATEGORY OUT OF DOCUMENTS OF EXTRACTED TOPICS ($\#D_{tc}$), $Precision = D_{tc}/D_t$, $Recall = D_{tc}/D$, $F - measure = 2 * Precision * Recall/(Precision + Recall)$

		$\#D$	$\#T$	$\#D_t$	$\#D_{tc}$	Precision	Recall	F-measure
requirements	general LDA	270	4	350	117	0.334	0.433	0.377
	our method		6	265	122	0.460	0.452	0.456
bug	general LDA	447	4	389	272	0.699	0.609	0.651
	our method		8	446	311	0.697	0.696	0.697
resource	general LDA	40	1	48	26	0.542	0.650	0.591
	our method		1	34	22	0.647	0.550	0.595

other hand, we can not expect much improvement to extract the category of the general function such as *resource*.

In keyword expansion by word2vec, it is considered that similar words could be extracted with high accuracy because semantically similar words have higher similarity. However, there are still semantically similar words that we cannot extract. In order to expand keyword, we could extract more general similar words by using full text of English Wikipedia as trainig data of word2vec. By using user reviews itself as training data of word2vec, there is a possibility that more similar words in reviews can be extracted.

VIII. CONCLUSIONS

In this paper, we proposed a method for extracting documents with contents required by developers with high accuracy. This method is based on topic classification by LDA and keyword expansion with word2vec. We applied this method to user reviews of a well-known application. Our experimental results demonstrated the validity of our method.

The results presented in this paper indicate some possible directions of further work and improvements. One of our primary on-going studies is further improvement of precision and recall rate of our extraction. We would like to establish a method for determining the appropriate number of topics. We also plan to define guidelines for keyword expansion. We will also conduct case studies using a large amount of user reviews to discover further findings for the extraction. We believe that automatic and sophisticated keyword expansion and topic classification help us to analyze user reviews efficiently.

REFERENCES

[1] "Techcrunch," https://techcrunch.com/2016/06/13/apples-app-store-hits-2m-apps-130b-downloads-50b-paid-to-developers/.

[2] C. Iacob and R. Harrison, "Retrieving and analyzing mobile apps feature requests from online reviews," in *Proceedings of the 10th Working Conference on Mining Software Repositories*, ser. MSR '13. Piscataway, NJ, USA: IEEE Press, 2013, pp. 41–44. [Online]. Available: http://dl.acm.org/citation.cfm?id=2487085.2487094

[4] E. Guzman and W. Maalej, "How do users like this feature? A fine grained sentiment analysis of app reviews," in *IEEE 22nd International Requirements Engineering Conference, RE 2014, Karlskrona, Sweden, August 25-29, 2014*, 2014, pp. 153–162. [Online]. Available: https://doi.org/10.1109/RE.2014.6912257

[3] 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. [Online]. Available: http://doi.acm.org/10.1145/2568225.2568263

[5] 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*, ser. KDD '13. New York, NY, USA: ACM, 2013, pp. 1276–1284. [Online]. Available: http://doi.acm.org/10.1145/2487575.2488202

[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. [Online]. Available: https://doi.org/10.1109/ICSE.2017.18

[7] C. E. Moody, "Mixing dirichlet topic models and word embeddings to make lda2vec," coRR, 2016.

[8] S. Li, T.-S. Chua, J. Zhu, and C. Miao, "Generative topic embedding: a continuous representation of documents," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2016, pp. 666–675. [Online]. Available: http://www.aclweb.org/anthology/P16-1063

[9] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Mar. 2003. [Online]. Available: http://dl.acm.org/citation.cfm?id=944919.944937

[10] 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*, ser. KDD '08. New York, NY, USA: ACM, 2008, pp. 569–577. [Online]. Available: http://doi.acm.org/10.1145/1401890.1401960

[11] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 3111–3119. [Online]. Available: http://dl.acm.org/citation.cfm?id=2999792.2999959

[12] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," coRR, 2013. http://arxiv.org/abs/1301.3781.

[13] "Mallet homepage," http://mallet.cs.umass.edu/.

[14] "Training word2vec model on english wikipedia by gensim," http://textminingonline.com/training-word2vec-model-on-english-wikipedia-by-gensim.

[15] "gensim: models.word2vec-deep learning with word2vec," https://radimrehurek.com/gensim/models/word2vec.html.

[16] H. Takahashi, H. Nakagawa, and T. Tsuchiya, "Towards automatic requirements elicitation from feedback comments: Extracting requirements topics using LDA," in *The 27th International Conference on Software Engineering and Knowledge Engineering, SEKE 2015, Wyndham Pittsburgh University Center, Pittsburgh, PA, USA, July 6-8, 2015*, 2015, pp. 489–494. [Online]. Available: https://doi.org/10.18293/SEKE2015-103