Opinion Analysis and Organization of Mobile Application User Reviews

Long Wang Hiroyuki Nakagawa Tatsuhiro Tsuchiya

Graduate School of Information Science and Technology,
Osaka University, Suita-shi, 565-0871, Japan
{l-wang, nakagawa, t-tutiya}@ist.osaka-u.ac.jp

Abstract

User reviews play a vital role in mobile application development. New users can grasp the pros and cons of an app from user reviews, and developers can improve the app by addressing the issues mentioned in the user reviews. However, scanning and analyzing massive user reviews is always a challenging and time-consuming task for both users and developers. This paper proposes a solution to build a tool for analyzing and organizing user reviews. To analyze user reviews, we classify the sentences in the reviews into predefined categories by using a machine learning algorithm; then, we apply text classification techniques to determine the review sentence's polarity and finally to mine key phrases from the sentences. We conducted an experiment using user reviews for two messaging apps. The experimental results demonstrate that we can organize the core information of each review and present the information to users and developers in respectively different ways.

1 Introduction

Mobile application development keeps growing fast today, and many applications become significantly important in our life. A popular application category, like messaging application, usually has more than one hundred apps available in the market. How to quickly find the right one among numerous apps can be difficult for new users. Fortunately, mobile application user reviews could provide necessary information for helping users to make the right decision [KM17] [SJ15]. Also, user reviews could help developers improve the application because the reviews report what kind of issues users have experienced or reveal why certain users like or dislike the application [KH06]. However, manually scanning massive unstructured reviews is always a challenge task [BGHM+08] [DLP03], since reviews usually contain many texts and require hours to browse.

Today many user reviews come typically with a rating that reflects the user's preference towards the application [Tan06], and these review ratings are intended to help users and developers speed up the review scanning process. However, since the review rating can be inconsistent and biased[DLP03], a single rating can not be a valuable medium to represent the entire application correctly. For example, a rating can be highly subjective when a user rated an app with one star when a minor feature failed. In another example, a user could write a review and

In: M. Sabetzadeh, A. Vogelsang, S. Abualhaija, M. Borg, F. Dalpiaz, M. Daneva, N. Fernández, X. Franch, D. Fucci, V. Gervasi, E. Groen, R. Guizzardi, A. Herrmann, J. Horkoff, L. Mich, A. Perini, A. Susi (eds.): Joint Proceedings of REFSQ-2020 Workshops, Doctoral Symposium, Live Studies Track, and Poster Track, Pisa, Italy, 24-03-2020, published at http://ceur-ws.org

give a perfect rating to an application at the beginning, but the user may find some issues afterward. If the user later modifies the review message without updating the rating, the perfect rating then will hide the potential issues.

The existing mobile application stores do not provide well-organized tools to assist people in scan reviews. In this study, we propose a tool for analyzing and organizing reviews, and therefore provide a solution to meet different kinds of demands from users and developers. When users and developers are scanning reviews, their focus could be different. Users like to browse a group of reviews together, which makes them focus on the collective ideas from a group. On the other hand, developers want to check the reviews successively, because each review could contain different issues with each other. With machine learning techniques, our tool is able to provide summarized information to users and developers in respectively different ways.

The rest of the paper is organized as follows: Section 2 sketches related work. Section 3 presents methods to design and construct the tool. Section 4 describes data collection process, and Section 5 explains our analyzing process. Section 6 demonstrates how we organize the results and present it to users and developers. Finally, Section 7 concludes with future work.

2 Related Work

Many studies demonstrated how machine learning techniques could help users and developers analyze and organize data. Rajeev et al. [RR15] and Priyanka et al. [PTB15] show solutions to advise users in finding online products. Chen et al. [CSM+18] present a tool to assist developers in software development. With practical examples, these studies show that machine learning could be a powerful means which helps users and developers to manipulate various kinds of data efficiently.

There has been a trend towards analyzing and organizing reviews with machine learning technologies. Many studies focus on text classification, data extraction, and information summarization. A standard method of analyzing user reviews is to determine whether a review presents a positive or negative attitude. Pang et al. [PLV02] and Turney et al. [Tur02] show that document-level sentimental analysis reaches a good result and even performs better than human-produced baselines. Tanawongsuwan et al. [Tan10] demonstrate how to process the sentiment classification on a product review through the analysis of parts of speech of the textual content. Instead of merely determining whether a review has a positive or negative tone, Maalej et al. [MN15] concentrate on review tagging. It proposed a method to determine whether a review is bug report, feature request, or praise.

Data extraction is also a pivotal process in analyzing reviews. Somprasertsri et al. [SL10] develop a method to extract product features and associated opinions from reviews through syntactic information based on dependency analysis. Dave et al. [DLP03] propose a method for extracting a product attribute and aggregating opinions about each of them; besides, this method can automatically differentiate positive reviews and negative reviews. Kim et al. [KH06] represent a method not only to extract a products' pros and cons from reviews but also to mine the sentences which account for the reviewer's preferences.

Beyond text classification and data extraction in user reviews, Hu et al. [HL04] focus on mining and summarizing reviews by extracting opinion sentences about product features. Blair-Goldensohn et al. [BGHM+08] concentrate on aspect-based summarization models, where a summary is produced by extracting relevant aspects of a local service, aggregating the sentiment per aspect, lastly selecting aspect-relevant text.

The previous studies mostly concentrated on analyzing reviews to help users. They target on-line store products and local service reviews, such as banks, restaurants, movies, and travel destinations. Our study differs in essential ways from previous ones: it focuses on mobile application user reviews. In recent years, there are also a number of studies concentrate on mining useful information from user reviews. Vu et al. [VNPN15] propose a framework to help analysts to collect and mine user opinions from reviews. Guzman et al. [GM14] propose an automated approach to help developers to filter and analyze user reviews. Our study is different as we are interested in building a tool that aims to analyze reviews for both users and developers.

3 Methodology

Our tool aims to analyze reviews and adequately organize them for users and developers. The overall goal of the tool is to help users and developers accelerate the review scanning process. In this study, we target on the sentence-level analysis, since each review may contain various opinions. Also, these opinions could cover different aspects of an app; for example, an opinion can discuss a specific function feature, or it can express a general option towards the entire app. Consequently, we target the sentence-level analysis and focus on analyzing user reviews in the following perspectives.

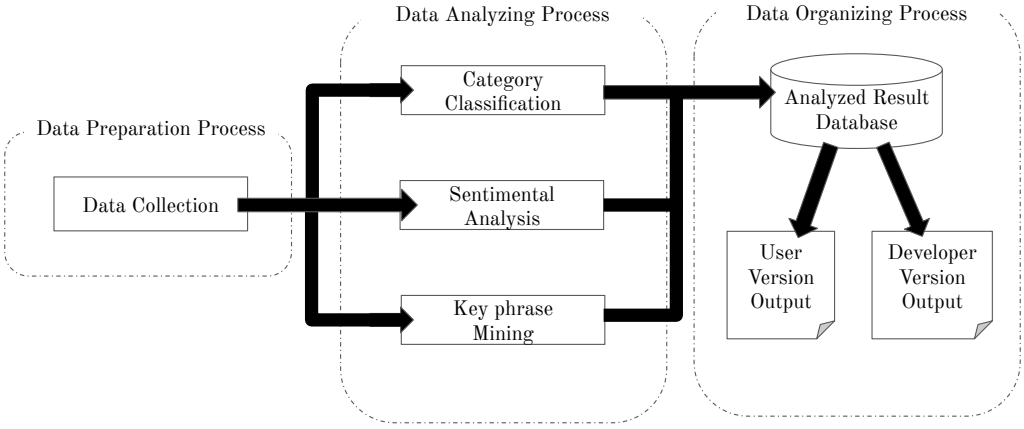

Figure 1: Workflow of the tool

- From what aspect did users review the app?
- What is the polarity of each review sentence?
- What is the primary meaning of a sentence?

Figure 1 shows our tool's workflow. The workflow consists of three major processes: the data preparation process, the data analyzing process, and the data organizing process. In the data preparation process, we crawl review data from the mobile application store and pre-process them for later analysis. In the data analyzing process, we employ three different kinds of machine learning techniques. We use text classification to classify each review sentence into a category tag. The category tag represents the review aspect, and thus solves the first of the above three problems. For the second problem, we use sentimental analysis techniques to generate the polarity tag for each review sentence. Lastly, in the key phrase mining process, we extract the core information from review sentences to solve the third problem. In the data organizing process, we store the analyzed data into a database and process them to provide output to users and developers.

4 Data Preparation

We use reviews from mobile messaging applications as examples to demonstrate the analyzing and organizing processes and tool implementation. Two popular messaging applications, WeChat[1] and LINE[2], are used in this study. They are available at the Google Play store and have sufficient user reviews. WeChat is a messaging and social media app - it is a lifestyle for one billion users across the world. It provides not only chatting and calling features but also gaming, mobile payment features, and many other features. In total, WeChat had 100, 000, 000+ installations and 5, 000, 000+ reviews. Another messaging application, LINE reshapes communication around the globe, allowing users to enjoy not only messaging but also free voice and video calls. LINE had 500, 000, 000+ installations and 11, 000, 000+ reviews.

Sometimes an app's update may introduce new bugs and unsatisfied feature changes, and then it can cause many alike user reviews. For the variety of data maintenance concerns, we collected review data from a one-year period (April 2018 to April 2019) and selected the first 100 English reviews from each month. In such a way, we can avoid similar bug reports or feature complaint reviews within a short period. Also, as suggested by Dave and Lawrence [DLP03], we filtered out the reviews, which contain less than three words or primarily written with symbols to avoid sparse data issue, from the collected data set.

[1] `https://play.google.com/store/apps/details?id=com.tencent.mm\&hl=en`
[2] `https://play.google.com/store/apps/details?id=jp.naver.line.android\&hl=en`

5 Data Analysis

Category classification. We employ text classification techniques to classify review sentences to category tags. In the classification, the pre-processed review textual content is used as the feature.

- **Review sentence tagging.** The tagging was performed manually in the category text classification process. The predefined categories for the messaging application reviews model include *General opinion*, *Functional feature*, and *Out of domain*. *General opinion* indicates a review sentence that has a broad opinion towards the application, and it generally presents whether users like the application or not. *Functional feature* indicates a review sentence that directly discusses a specific function or a feature of the application. *Out of domain* indicates a phrase that is neither talking about a specific function or a general review, and it describes a type of phrase that usually contains a single word or an unrelated discussion. Both WeChat and LINE's review data are annotated with the above three categories. Table 1 shows the total number of sentences, the numbers of sentences tagged with different categories, and the number of vocabularies appearing in the review data.

- **The text pre-processing.** Considering that users might write non-standard English words, such as emoji and online slang, in reviews, we simplify and clean the text before training. In our study, the processing steps are: lowering case conversion, converting numbers to words, tokenizing sentence, removing stop words, and reducing word forms. We use Natural Language Toolkit (NLTK) [3] library to break a sentence into words; this library has a Tweet Tokenizer module which can recognize online slang and diminishes the word length; for example, "waaaaayyyy" is reduced to "waaayyy".

- **Convolutional neural networks.** In this study, the text classification algorithm is based on character-level Convolutional Neural Networks (CNN). Recent studies proved that CNN works well for problems in the natural language processing [Kim14] [ZZL15], although CNN is more often apply to solve machine learning image problems. Gradually the CNN model had become a standard baseline for new text classification architectures. Unlike image pixels that are used as input in image problems, the input in our classification problem is a review sentence represented as matrix. Each row of the matrix is a vector that represents a word; moreover, the vector is the index of a word into vocabularies appearing in our collected data.

Table 1: **Statistics For Sentences Labeling**

Categories	**WeChat**	**LINE**	**Combined**
# Sentences	3417	2803	6220
# Functional feature sentences	1486	1007	2493
# General opinion sentences	1034	1030	2063
# Out of domain sentences	897	766	1664
# Vocabularies	3026	2744	4248*
# Preprocessed vocabularies	2127	1974	2955*

(Note that duplicates were removed in the number of vocabularies and the number of preprocessed vocabularies when combining vocabularies from WeChat and LINE.)

Category classification cross-validation. To experimentally evaluate the performance of the category classification process, we conducted a 3-fold cross validation using the reviews we collected. The whole data set was shuffled and equally divided into three parts in the evaluation process. Each set was used as the testing set once, and the other remaining two sets were used as the training set. As a result, three classification models were developed for a given collection of data in the experiment. Table 2 presents classification results of using only WeChat data, Table 3 presents the classification results of using only LINE data, and Table 4 shows the classification results of using data from both WeChat and LINE.

All the tables show that classification models have satisfying prediction scores on these three categories. The *functional feature* category has the best results, and the *out of domain* has the least prediction rate. The reason could be that the *functional feature* has the most extensive training set size, whereas the *out of domain* has the least. Although WeChat had a slightly smaller number of reviews, each review from WeChat seems to

[3] available at `https://www.nltk.org/`

Table 2: Evaluation result for WeChat

Categories	**Precision**	**Recall**	**F1-score**	**No. Test**
Functional feature (1)	0.76	0.81	0.78	448
General opinion (1)	0.72	0.69	0.71	304
Out of domain (1)	0.63	0.59	0.61	249
Functional feature (2)	0.81	0.84	0.83	453
General opinion (2)	0.78	0.70	0.74	298
Out of domain (2)	0.61	0.64	0.62	250
Functional feature (3)	0.78	0.82	0.80	428
General opinion (3)	0.73	0.74	0.73	305
Out of domain (3)	0.64	0.58	0.61	267
Functional feature (Avg)	**0.783**	**0.823**	**0.803**	**443**
General opinion (Avg)	**0.743**	**0.710**	**0.727**	**302**
Out of domain (Avg)	**0.627**	**0.603**	**0.613**	**255**

Table 3: Evaluation result for LINE

Categories	**Precision**	**Recall**	**F1-score**	**No. Test**
Functional feature (1)	0.76	0.79	0.78	294
General opinion (1)	0.65	0.70	0.68	302
Out of domain (1)	0.68	0.59	0.63	251
Functional feature (2)	0.82	0.79	0.80	303
General opinion (2)	0.66	0.81	0.73	318
Out of domain (2)	0.68	0.49	0.57	227
Functional feature (3)	0.71	0.76	0.74	313
General opinion (3)	0.70	0.64	0.67	327
Out of domain (3)	0.57	0.58	0.57	207
Functional feature (Avg)	**0.763**	**0.78**	**0.773**	**294**
General opinion (Avg)	**0.670**	**0.717**	**0.693**	**316**
Out of domain (Avg)	**0.643**	**0.553**	**0.590**	**228**

contain more sentences and generate a large data set. As a result, the WeChat classification model generally performs better than the LINE model. Since the same tagging rules were applied to WeChat and LINE review data, we used the mixed data together to train another classification model. Appropriately, this classification model also ended up satisfying classification results.

Sentiment analysis. After determining a review sentence category, we examine each sentence on whether its attitude is positive or negative. For instance, consider the following review examples, "WeChat send messages fast" and "WeChat lagging on taking pictures." They both discuss functional features, but they have very different attitudes towards the features. Knowing a sentence's polarity undoubtedly helps users and developers accelerate the scanning process. In this study, we use the NLTK library again for sentiment classification. It can determine a review sentence, whether it expresses positive sentiment, negative sentiment, or if it is neutral. This library uses hierarchical classification, where a sentence's neutrality is checked first, and then the polarity is determined.

Key phrases mining. To help developers speed up to discover any issue from reviews, we reduce the text content amount developers need to read in reviews. To this end, we mine key phrases from review sentences. We use a library called RAKE-NLTK (RAKE stands for Rapid Automatic Keyword Extraction algorithm [4]), to mine key phrases. RAKE is a domain-independent keyword extraction algorithm, and it tries to discover key phrases from the text body by analyzing the frequency of word appearance and word's co-occurrence with other

[4]available at `https://github.com/csurfer/rake-nltk`

Table 4: Evaluation result for mixed data

Categories	Precision	Recall	F1-score	No. Test
Functional feature (1)	0.80	0.77	0.79	806
General opinion (1)	0.71	0.69	0.70	682
Out of domain (1)	0.59	0.65	0.62	511
Functional feature (2)	0.76	0.79	0.78	811
General opinion (2)	0.69	0.70	0.69	652
Out of domain (2)	0.62	0.57	0.60	537
Functional feature (3)	0.74	0.82	0.78	806
General opinion (3)	0.71	0.66	0.68	672
Out of domain (3)	0.62	0.58	0.60	522
Functional feature (Avg)	**0.767**	**0.793**	**0.783**	**808**
General opinion (Avg)	**0.703**	**0.683**	**0.690**	**669**
Out of domain (Avg)	**0.610**	**0.600**	**0.607**	**523**

words in the text [RECC10]. The mining algorithm takes a text sentence as an input and outputs a list of text phrases with related scores. In this study, a simple filter algorithm was applied to the mining results. We rank the results by their scores from the highest to lowest, and we filter out the phrases whose scores are equal to the lowest score in the results.

6 Data Organization

In this section, we describe how the analyzed information is presented into two kinds of output for users and developers.

The user version. This version of output is organized to help new users to quickly answer the following two questions: 1) From what aspect did other users review the app? and 2) What are the polarities of these aspects? For this reason, we select the review category tag and the polarity tag information from the analyzed result database and draw them in the form of bar graphs for users. Figure 2 shows an example of analyzed WeChat reviews from April 2019. The graph illustrates the number of aspects, which are *out of domain*, *general opinion*, and *functional feature*, mentioned in the reviews. The polarity information is colored and visibly displayed. More importantly, the original review sentence can be traversed by clicking the corresponding chart bar. Additionally, user version output supports the comparison mode so that users can compare the analyzed reviews from two apps. Figure 3 shows an example of comparing results between WeChat and LINE; in the figure, two analyzed results are horizontally presented side by side in one output. The output can efficiently show the difference to users, and therefore clearly present which app is preferred by users in a given period of time.

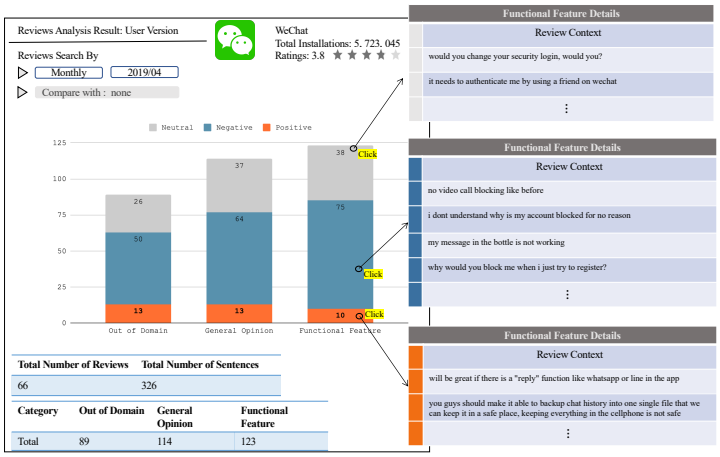

Figure 2: Example of user version output displaying WeChat reviews

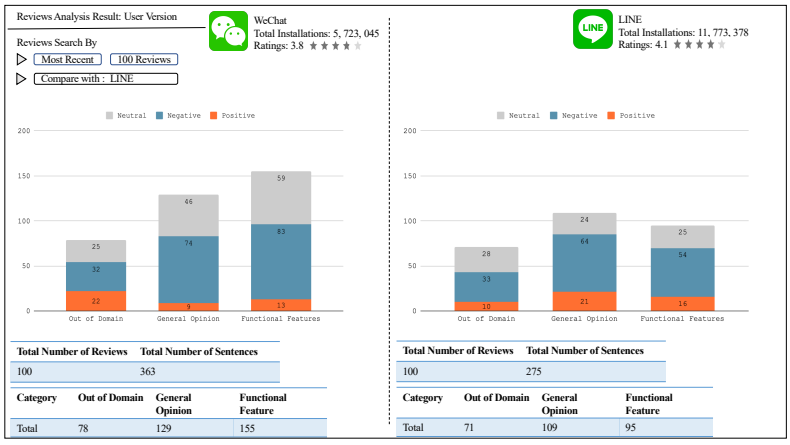

Figure 3: User version output comparing most recent 100 WeChat and LINE reviews

The developer version. This version of output aims to assist developers to discover what kind of issue users have when using the app. For this, each review is transformed into a compact version in the output. The review sentence is replaced with a category tag, a corresponding polarity tag, and a list of key phrases. The polarity tag combined with the list of key phrases can reveal the potential issues users mentioned in the review, and the category tag indicates what aspect the issues mentioned. Similar to the user version's output, the original review text can be traversed by clicking the corresponding tags. Figure 4 shows an example of developer version output for WeChat, and Figure 5 shows the difference between an original review and the related developer version output. The developer version output reduced many texts from the original review, but it still retains the core information. By reading the output, developers are able to develop an abstract understanding of the original review.

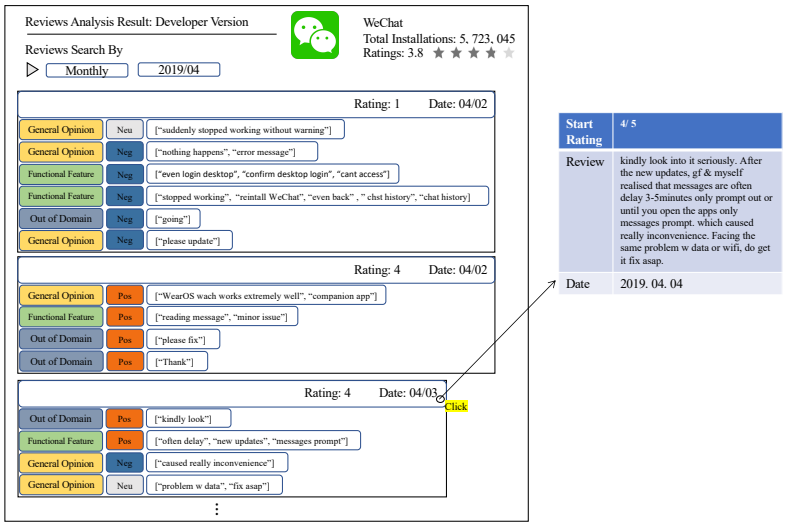

Figure 4: Example of developer version output displaying WeChat reviews

7 Conclusion and Future Work

In this paper, we proposed a method to build a tool for analyzing and organizing user reviews. This tool can generate two kinds of output in order to meet different demands from users and developers. The user version

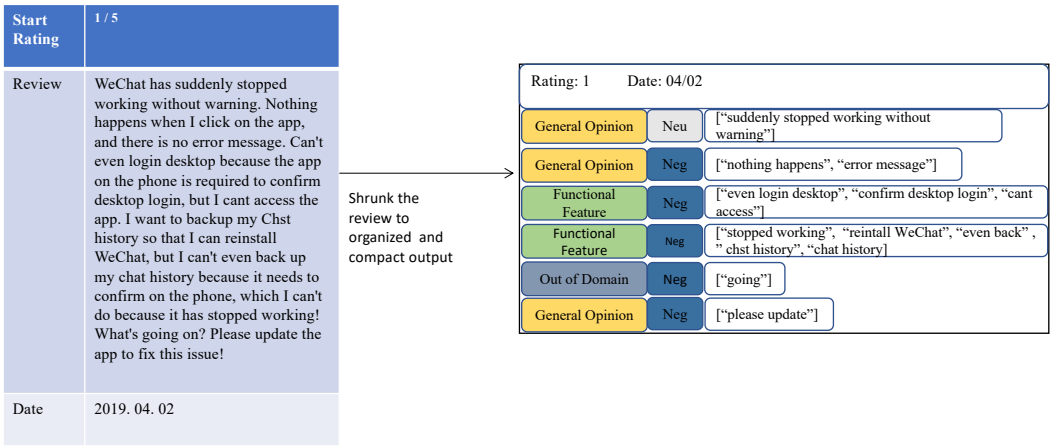

Figure 5: A comparison between the developer version output and the original review. The developer version output significantly reduces the text amount and requires less scanning time.

output focuses on presenting the character of a group of data, and the developer version output focuses on shrinking the review texts. Our tool consists of two primary parts: using machine learning techniques to analyze user reviews, and organizing analyzed results for users and developers separately. We keep working on the following tasks to build a more sophisticated tool in the future.

In future work, an actual human evaluation should be involved. We will give the generated outputs to real developers and ordinary volunteers, and ask them to provide confident rates for the analyzed and organized results. Another work could be to improve the accuracy of key phrase mining. Currently, the algorithm of filtering the mining result is naive. Additional techniques, such as removing the stop words and unnecessary adjectives, could make the mining results clearer and more reliable.

References

[BGHM+08] Sasha Blair-Goldensohn, Kerry Hannan, Ryan McDonald, Tyler Neylon, George Reis, and Jeff Reynar. Building a sentiment summarizer for local service reviews. In *WWW Workshop on NLP Challenges in the Information Explosion Era (NLPIX)*, 2008.

[CSM+18] Chunyang Chen, Ting Su, Guozhu Meng, Zhenchang Xing, and Yang Liu. From ui design image to gui skeleton: A neural machine translator to bootstrap mobile gui implementation. In *Proceedings of the 40th International Conference on Software Engineering*, ICSE '18, pages 665–676, New York, NY, USA, 2018. ACM.

[DLP03] Kushal Dave, Steve Lawrence, and David Pennock. Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. *Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews*, 775152, 10 2003.

[GM14] 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.

[HL04] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In Won Kim, Ron Kohavi, Johannes Gehrke, and William DuMouchel, editors, *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22-25, 2004*, pages 168–177. ACM, 2004.

[KH06] Soo-Min Kim and Eduard H. Hovy. Automatic identification of pro and con reasons in online reviews. In *ACL*, 2006.

[Kim14] Yoon Kim. Convolutional neural networks for sentence classification. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751. ACL, 2014.

[KM17] Z. Kurtanović and W. Maalej. Mining user rationale from software reviews. In *2017 IEEE 25th International Requirements Engineering Conference (RE)*, pages 61–70, Sep. 2017.

[MN15] W. Maalej and H. Nabil. Bug report, feature request, or simply praise? on automatically classifying app reviews. In *2015 IEEE 23rd International Requirements Engineering Conference (RE)*, pages 116–125, Aug 2015.

[PLV02] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 79–86. Association for Computational Linguistics, July 2002.

[PTB15] K. Priyanka, A. S. Tewari, and A. G. Barman. Personalised book recommendation system based on opinion mining technique. In *2015 Global Conference on Communication Technologies (GCCT)*, pages 285–289, April 2015.

[RECC10] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. *Automatic Keyword Extraction from Individual Documents*, pages 1 – 20. 03 2010.

[RR15] P. V. Rajeev and V. S. Rekha. Recommending products to customers using opinion mining of online product reviews and features. In *2015 International Conference on Circuits, Power and Computing Technologies [ICCPCT-2015]*, pages 1–5, March 2015.

[SJ15] Ananthi Sheshasaayee and R. Jayanthi. A text mining approach to extract opinions from unstructured text. *Indian Journal of Science and Technology*, 8(36), 2015.

[SL10] G. Somprasertsri and P. Lalitrojwong. Extracting product features and opinions from product reviews using dependency analysis. In *2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery*, volume 5, pages 2358–2362, Aug 2010.

[Tan06] P. Tanawongsuwan. Relation between a book review content and its rating. In *International Conference on Computer Information Systems and Industrial Applications*. Atlantis Press, 2015/06.

[Tan10] P. Tanawongsuwan. Product review sentiment classification using parts of speech. In *2010 3rd International Conference on Computer Science and Information Technology*, volume 8, pages 424–427, July 2010.

[Tur02] Peter D. Turney. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. *CoRR*, cs.LG/0212032, 2002.

[VNPN15] P. M. Vu, T. T. Nguyen, H. V. Pham, and T. T. Nguyen. Mining user opinions in mobile app reviews: A keyword-based approach (t). In *2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 749–759, Nov 2015.

[ZZL15] Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 649–657, 2015.