Deficient Documentation Detection

A Methodology to Locate Deficient Project Documentation using Topic Analysis

Joshua Charles Campbell*, Chenlei Zhang*, Zhen Xu†, Abram Hindle*, James Miller†
*Department of Computing Science
†Department of Electrical and Computer Engineering
University of Alberta
Edmonton, Canada
{joshua.campbell, chenlei.zhang, zxu3, abram.hindle, james.miller}@ualberta.ca

***Abstract*—A project's documentation is the primary source of information for developers using that project. With hundreds of thousands of programming-related questions posted on programming Q&A websites, such as Stack Overflow, we question whether the developer-written documentation provides enough guidance for programmers. In this study, we wanted to know if there are any topics which are inadequately covered by the project documentation. We combined questions from Stack Overflow and documentation from the PHP and Python projects. Then, we applied topic analysis to this data using latent Dirichlet allocation (LDA), and found topics in Stack Overflow that did not overlap the project documentation. We successfully located topics that had deficient project documentation. We also found topics in need of tutorial documentation that were outside of the scope of the PHP or Python projects, such as MySQL and HTML.**

***Index Terms*—Stack Overflow, documentation, LDA, topic analysis**

I. Motivation

Software engineers often have to rely on language or project documentation to solve programming problems, yet project documentation is not always sufficient or clear enough to actually help software engineers. Fortunately, programmers can find crowd-sourced help online instead of relying on project documentation. There are many websites that allow users to ask and answer technical questions, such as Stack Overflow,[1] Ask Ubuntu,[2] Super User,[3] and CSDN BBS.[4] Many different kinds of questions are asked by users at all levels of expertise.

In this paper we focus on Stack Overflow questions about PHP and Python. We make the three contributions: 1) we provide a semi-automatic method to relate crowd-sourced questions and project documentation; 2) we answer the question, "Can we identify deficient areas of project documentation by relating it to Stack Overflow questions?"; 3) we provide a method to locate deficient project documentation.

II. Related Work

As surveys performed by Lethbridge *et al.* [1] reveal, software project documentation is often out of date, poorly written or incomplete. The related work in this topic is organized into three categories: works that focus on developer-written project documentation, works that focus on crowd-sourced documentation mined from the Internet including websites such as Stack Overflow, and works related to the specific techniques employed in this paper.

Previous research on developer-written project documentation quality was surveyed in Garousi's thesis [2]. The techniques surveyed include automated readability measurement and perceived quality of project documentation studies. Automated readability measurement techniques do not take into account any sources of data other than the project's documentation itself. Other authors, such as Forward *et al.* [3] focused on perceived quality of project documentation by surveying developers manually. Another technique, that was suggested by del Galdo *et al.* [4], requires developers to manually report deficiencies in the project's documentation when they discover them.

Parnin *et al.* [5] discussed the use of Stack Overflow as a primary source for API documentation and measured the incompleteness of Stack Overflow in that role. We, conversely, measured the incompleteness of developer-written project documentation. Parnin *et al.* [5] and Nasehi *et al.* [6] analyzed code examples from Stack Overflow. However, they considered crowd-sourced, user-generated code examples exclusively, and not those provided by project developers and documenters. Parnin *et al.* [7] also analyzed API documentation specifically, but again only considered crowd-sourced, online documentation. They concluded that such documentation typically does not completely cover an API, finding a maximum of 87.9% coverage. Jiau *et al.* [8] also discussed the lack of complete and equitable coverage when using only crowd-sourced documentation. Finally, Kononenko *et al.* [9] provided a tool to automatically locate crowd-sourced documentation inside an integrated development environment (IDE).

In our study, we combined both crowd-sourced documentation and developer-written project documentation, and applied a topic model to them. Topic models, such as latent Dirichlet allocation (LDA) [10] have been widely applied in software engineering for the purpose of understanding software systems and linking their artifacts together. Lukins *et al.* [11] retrieved source code with LDA-based analysis techniques to locate

[1]Stack Overflow: http://stackoverflow.com/
[2]Ask Ubuntu: http://askubuntu.com/
[3]Super User: http://superuser.com/
[4]CSDN BBS: http://bbs.csdn.net/

978-1-4673-2936-1/13 © 2013 IEEE

MSR 2013, San Francisco, CA, USA

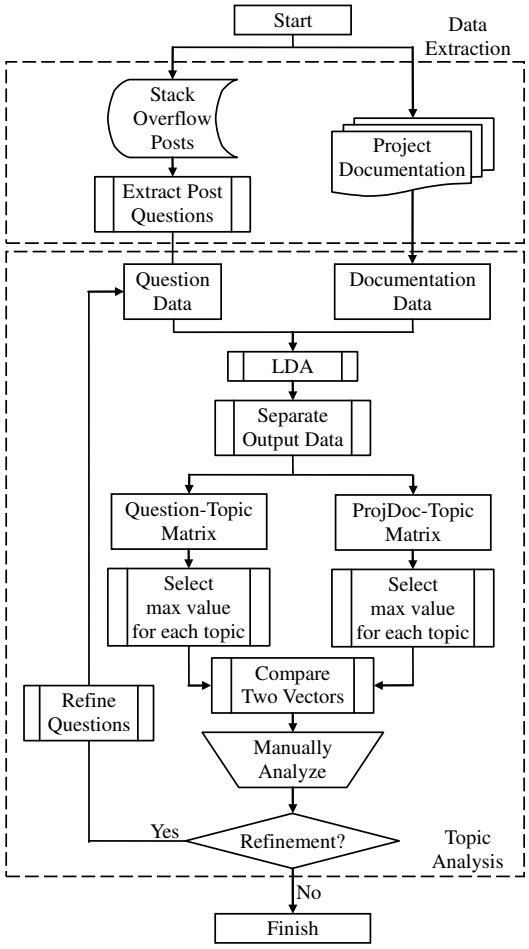

Figure 1. Flow chart presentation of our methodology.

bugs. Linstead *et al.* [12] automatically generated traceability links for artifacts in software projects by applying LDA. Barua *et al.* [13] analyzed the relationships and trends of the main topics discovered by LDA from posts in Stack Overflow.

III. Methodology

To compare user-generated questions and official project documentation, we first extracted textual information from both sources. Then, in the second step, we used LDA to analyze the combined data from both sources. LDA's output indicates which posts and documentation discuss each topic. By analyzing the posts and documentation discussing a topic, we located deficient project documentation.

A. Data Extraction

The data sources used in our implementation are: 1) the Stack Overflow website's post data provided by MSR 2013 [14]; 2) the PHP project's official offline documentation,[5] which does not include user comments, and 3) the Python project's official offline documentation.[6]

In order to extract the Stack Overflow post data, we used the Java SAX parser to locate all questions. Votes, answers, badges, and user ratings were not considered. We retrieved post identification numbers, post titles, post tags and post contents as plain text. Because we investigated PHP and Python, we only kept posts tagged with "PHP" or "Python."

For the project documentation, we extracted the text of the documentation that developers would read in each file for PHP and Python. Additionally, we assigned each piece of project documentation an identification number that is distinct from the identification numbers of Stack Overflow questions.

After combining the two sources together for each project, we used the ScalaNLP API[7] to separate words by space and punctuation, remove non-words and non-numbers, remove words less than two characters and remove standard English stop words.

B. Topic Modelling

We separately applied LDA to the combined Stack Overflow and project documentation data for PHP and Python using the CVB0 algorithm implemented in the Stanford Topic Modelling Toolbox [15]. LDA requires an input parameter K, that is the number of topics that LDA will extract. In our implementation we chose the highest number of topics that we could, $K = 400$, limited only by the available RAM and CPU time, to produce more detailed topics [13]. Additionally, we set the LDA hyper-parameters α and β to be 0.01. The result of LDA is a matrix M where rows correspond to the Stack Overflow questions or project documentation and columns correspond to K topics.

C. Topic Analysis

In order to analyze the topics, first the LDA output matrix M was split into two pieces. If the document identifier was from Stack Overflow, the document's LDA results were placed in the Stack Overflow matrix, otherwise they were placed in the project documentation matrix.

Table I
Topics from PHP with Largest Differences

Topic	Top 10 Terms	Topic	Top 10 Terms
311	pdo ingres transaction ibase driver pdostatement transactions database dsn drivers	180	company complex node.js metadata companies department sugarcrm crm leads lead
130	ajax jquery lost captcha php recaptcha verification dojo xmpp lose	258	htaccess rewrite mod mod-rewrite apache file php _rewrite url urls

Next, the two split matrices were analyzed. For each topic, the maximum correlation was selected for both the Stack Overflow questions and project documentation. Then, all topics were sorted by the difference of these two maxima. A typical result is presented in Figure 2.

The sorted maximum differences contains, at its beginning (left side of Figure 2), the largest differences which represent poorly documented topics. The 4 PHP topics with the largest differences are listed in Table I. Large-difference topics are followed by topics with small differences, which represent well-documented topics, and, finally, negative differences (right side of Figure 2), which represent rarely asked about topics.

[5]PHP Documentation: http://www.php.net/download-docs.php
[6]Python Documentation: http://docs.python.org/3/
[7]Scala NLP Breeze: http://www.scalanlp.org/

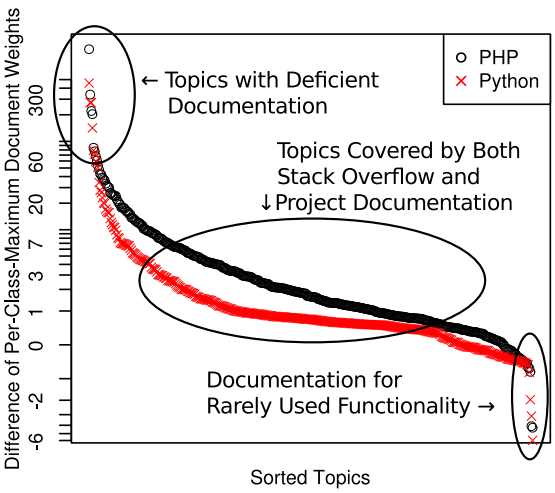

Figure 2. Distribution of Per-Class-Maximum Document Weight Differences. The Y-axis represents the difference between the most representative Stack Overflow post and the most representative project documentation document. The X-axis shows all topics sorted by this difference. PHP and Python were processed completely independently.

D. Manual Analysis

The final step required analyzing Stack Overflow posts and project documentation manually. This was simply a matter of reading the top-correlating questions. The top-correlating questions can be found by sorting the column for a topic in the question-topic matrix, and then retrieving the top-ranked questions from Stack Overflow.

Then, we decided if each post contained a question that should be documented by the project documentation authors. If so, we looked for relevant project documentation, either by Internet search or by retrieving the project documentation with highest LDA document-topic weight for that topic.

E. Input Refinement

Many of the top results produced by this method were not within the project's documentation's scope or responsibility. Therefore, we iteratively refined our input to remove questions regarding these topics. Refinement is accomplished by removing questions with tags containing specific strings from the initial LDA input. Then, we re-applied LDA and redid all of the subsequent analysis.

For example, the top topic for PHP was initially related to WordPress, which is a blogging platform written in PHP. We removed all the questions tagged with "WordPress" from the input data set. Then we applied LDA, topic analysis, and manual analysis steps as described above.

IV. Results and Discussion

All of the top 10 LDA topics that are discussed on Stack Overflow and that are not covered well in the project documentation (Figure 2, left side), and their top-weighted posts were examined. These topics fall into two categories: topics that should be covered by the project documentation that are either not covered or are covered deficiently, and topics that are outside of the scope of the project documentation and are therefore not covered there. Unfortunately, the line between these two categories is not always clear.

Most of the results are out-of-scope questions before input refinement. These questions have a large impact on our method, and make the input refinement step necessary. If no input refinement is performed, out-of-scope questions will crowd out the in-scope deficiently documented questions we are looking for.

Questions with high topic weight for well-documented topics (found in the middle of Figure 2) are typical Stack Overflow questions about bugs, algorithms, or syntax errors. Project documentation pages with high topic weight for rarely asked about topics (found on the right-side of Figure 2) are usually about project features which are not used often such as Python's deprecated RFC 2822 mail header parser.

A. Deficient Project Documentation

As an example of the first type of result, Stack Overflow question #7321289, was found using our method. This question asks:

> "How [to or I] want to apply a vignette effect to an image using PHP with ImageMagik. I found this function but I'm not sure how to use it."

The question then goes on to link directly to the `ImageMagick::vignetteImage` page in the PHP manual. After reading that manual page, none of the authors of this paper could understand how to use the function. Clear statements about the units of the input variables, valid and typical values, and a code example that performs a basic vignette are clearly required.

Another example is Stack Overflow question #6956861, which asks about installing PHP for use with the Apache Tomcat webserver. This is not documented by PHP at all, though PHP does include instructions on how to use PHP with the Apache Apache webserver. We discovered, by searching the web, that there are instructions on Tomcat's wiki about how to use PHP with Tomcat. It is not clear that documenting Tomcat integration is PHP's responsibility. It may not be reasonable to expect PHP to document the installation of PHP for use with every webserver. However, is it the responsibility of the Tomcat project to document the installation process of every language for use with Tomcat?

In Stack Overflow question #9219795, the user wishes to remove a certain number of characters from the end of a string, and has found the `substr()` function, which is certainly appropriate for doing so. PHP's project documentation includes an example of `substr()` usage that removes the last character from the end of a string. However, this example is not explained. It's only commentary contains an example input and output, but what it does is never mentioned explicitly. Merely adding "remove the last character from a string" to the comment on the example would be helpful.

Question #1781549 asks how to get the size of an image file in PHP. Of course, one would use the `stat()` function to obtain file meta-data such as size, but the PHP documentation for `stat()` only mentions that it returns file size as one of the

results. All of the code examples in the PHP documentation are for other uses of `stat()`, such as retrieving the date that the file was modified on.

One example of a Python coding topic that comes up frequently in the top results, regards the use of the _ (underscore) variable name. See, for example, question #5893163, where a user wonders what _ means. Using _ to identify variables which are set but never used in Python is apparently common practice, but not documented anywhere in the Python project documentation. In fact, this practice is recommended often on Stack Overflow.

B. Out-of-Scope Results

All of the previous examples occur within the top 10 topics (topics on the left of Figure 2) and the 5 posts with the highest weight for those topics. However, there are also many results that might be considered noise by a project documentation author. These topics are typically legitimate answers to the question, "What do people want to know about my project that isn't included in the project documentation?" In the case of questions regarding PHP, one common subject of results is SQL, blogging software such as WordPress, templating systems such as CakePHP, and, of course, HTML. These topics are asked about on Stack Overflow frequently and not documented by the PHP project.

For some of these projects, such as CakePHP and WordPress, that are built on top of PHP, it is clear that their project documentation is not the responsibility of the PHP project documentation authors. However, for some other domains it is not as clear. Two of these stand out for PHP: SQL and HTML. PHP scripts commonly invoke SQL to fetch data, and PHP is an HTML templating language.

It became clear to us as we examined the results of our technique that this is indeed an issue that needs to be addressed in project documentation and the software development community as a whole. Software depending upon multiple projects is commonplace. It is often unclear to developers which project's documentation should be examined to answer their question. Project documentation should provide tutorials that address these situations.

> Based on our results, we suggest that a project's documentation include clear indications and links when a user should reference external project documentation, such as the documentation for HTML or MySQL.

A particularly good example of this was found in Stack Overflow question #10474179 where a user asks whether one should use PHP or MySQL to limit the number of rows in a MySQL table to 500. These rows are inserted by PHP code. Indeed, the authors of this paper cannot agree on the answer to that question, because there are many possible solutions.

A common task in PHP is building templating systems, because PHP is an HTML templating language. Therefore, PHP project documentation should include recipes, examples and best practices for building templating systems in PHP. This would help address Stack Overflow questions such as #5629853 and other questions highlighted by our approach.

V. Conclusion

We have developed, implemented, and presented a method for locating aspects of a project that are inadequately documented by combining data from Stack Overflow and the project's documentation. This method uses LDA to guide the manual analysis of topics that may not have been sufficiently addressed by the project's documentation. In addition, it provides example questions that motivate the improvement of the project documentation.

> This semi-automatic method suggests topics whose project documentation could be improved.

Our method successfully locates areas of deficient project documentation using Stack Overflow questions, and we provide some relevant examples. However, these areas were not always clearly the project's responsibility to document.

References

[1] T. Lethbridge, J. Singer, and A. Forward, "How Software Engineers Use Documentation: The State of the Practice," *Software, IEEE*, vol. 20, no. 6, pp. 35 –39, nov.-dec. 2003.

[2] G. Garousi, "A Hybrid Methodology for Analyzing Software Documentation Quality and Usage," Ph.D. dissertation, University of Calgary, 2012.

[3] A. Forward and T. C. Lethbridge, "The Pelevance of Software Documentation, Tools and Technologies: A Survey," in *Proceedings of the 2002 ACM symposium on Document engineering*, ser. DocEng '02. New York, NY, USA: ACM, 2002, pp. 26–33.

[4] E. M. del Galdo, R. C. Williges, B. H. Williges, and D. R. Wixon, "An Evaluation of Critical Incidents for Software Documentation Design," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 30, no. 1, 1986, pp. 19–23.

[5] C. Parnin, C. Treude, L. Grammel, and M.-A. D. Storey, "Crowd Documentation: Exploring the Coverage and the Dynamics of API Discussions on Stack Overflow," Georgia Tech, Technical Report GIT-CS-12-05, 2012.

[6] S. M. Nasehi, J. Sillito, F. Maurer, and C. Burns, "What Makes a Good Code Example?: A Study of Programming Q&A in StackOverflow," in *ICSM*, 2012, pp. 25–34.

[7] C. Parnin and C. Treude, "Measuring API Documentation on the Web," in *Proceedings of the 2nd international workshop on Web*, vol. 2, 2011, pp. 25–30.

[8] H. C. Jiau and F.-P. Yang, "Facing up to the Inequality of Crowdsourced API Documentation," *SIGSOFT Softw. Eng. Notes*, vol. 37, no. 1, pp. 1–9, Jan. 2012.

[9] O. Kononenko, D. Dietrich, R. Sharma, and R. Holmes, "Automatically Locating Relevant Programming Help Online," in *VL/HCC*, 2012, pp. 127–134.

[10] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet Allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Mar. 2003.

[11] S. Lukins, N. Kraft, and L. Etzkorn, "Source Code Retrieval for Bug Localization Using Latent Dirichlet Allocation," in *Reverse Engineering, 2008. WCRE'08. 15th Working Conference on*. IEEE, 2008, pp. 155–164.

[12] E. Linstead and P. Baldi, "Mining the Coherence of GNOME Bug Reports with Statistical Topic Models," in *Proceedings of the 2009 6th IEEE International Working Conference on Mining Software Repositories*, ser. MSR '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 99–102.

[13] A. Barua, S. W. Thomas, and A. E. Hassan, "What are Developers Talking About? An Analysis of Topics and Trends in Stack Overflow," *Empirical Software Engineering*, p. To appear, 2012.

[14] A. Bacchelli, "Mining Challenge 2013: Stack Overflow," in *The 10th Working Conference on Mining Software Repositories*, 2013, p. to appear.

[15] D. Ramage and E. Rosen, "Stanford Topic Modeling Toolbox," Dec. 2011. [Online]. Available: http://nlp.stanford.edu/software/tmt/tmt-0.4