International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169
Volume: 4 Issue: 4 181 - 185

Extracting Tasks from Customize Portal using Natural Language Processing

Prachi Rayate
PG. Student, Dept. of Computer Engg.
Bharati Vidyapeeth Deemed University College of Engg.
Pune, India
e-mail:prachirayate@gmail.com

Prof. Devendra Singh Thakore
H.O.D. Dept. of Computer Engg.
Bharati VidyapeethDeemed University College of Engg
Pune, India
e-mail:dmthakore@bvucoep.edu.in

Abstract— In software documentation, product knowledge and software requirement are very important to improve product quality. Within maintenance stage, reading of whole documentation of large corpus won't be possible by developers. They need to receive software documentation i.e. (development, designing and testing etc.) in a short period of time. Important documents are able to record in software documentation. There live a space between information which developer wants and software documentation. To solve this problem, an approach for extracting relevant task that is based on heuristically matching the structure of the documentation under four phases of software documentation (i.e. documentation, development, testing and other) is described. Our main idea is that task is extracted automatically from the software documentation, freeing the developer easily get the required data from software documentation with customize portal using WordNet library and machine learning technique. And then the category of task can be generated easily from existing applications using natural language processing. Our approach use WordNet library to identify relevant tasks for calculating frequency of each word which allows developers in a piece of software to discover the word usage.

Keywords- Natural language processing, text mining, software documentation, machine learning, WordNet library etc.

*****

I. INTRODUCTION

The ability of the system to process sentences in a natural language such as English is called as "Natural language processing". To improve quality of product software and product knowledge these two are very crucial components. Data delivery to developers is the main objective of the important document. There are many forms which are covered in software documentation and these forms are wanted by software developer [3]. Many organizations of software development and open source projects try to solve the problem of space between software documentation and the information which developer wants by creating web pages which generated very useful information.

During maintenance work developer may not get enough time to read documentation of large systems also one time reading does not able to clear all definitions and motivation of the documentation. As the whole document consist of large information the statistical technique, automatically of task structure generally does not prove very useful. As per the user requirements search engines are inadequate and unable to express techniques. The technique gap is reduced by completing the words by search engines which have a software function to do the same and it is presented by high fulfilled user's feedback [4]. For customized search systems the count of previous appeared queries is not too large and also query logs are not available to learn such models [6].

The knowledge of pre-defined require to carry out the same task and that we have selected to use is Word Net, a lexical database consisting of a number of cross-referenced binary relationships on many nouns and verbs.

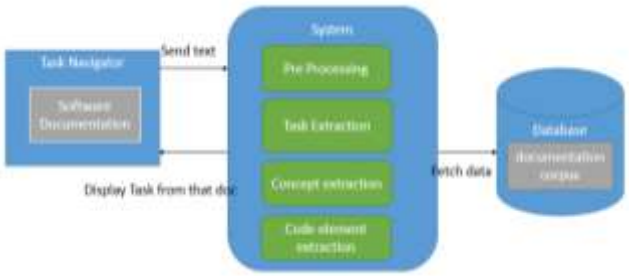

Fig 1: Task extraction process

WordNet contains information about a number of different types of relationships between words. The correctness of the processing steps is calculated. Using a standard of sentences and their equivalent tasks, we implemented the task extraction algorithm. With the help software developers and also calculated an evaluation of the tasks extracted from the documentation of customized portal to evaluate whether the extracted tasks are meaningful to developers. The result shows that it reduced a live space between information which developers want and software documentation automatically. The better results are evaluated by taking feedback from developers in terms of accuracy.

The main contributions of this paper are:
As the data stored into database whenever we want it for getting task, for software documentation, for preparing KDD and other purposes as well.

1. This work is mainly focus on software documentation to extract some task from large corpus through WordNet library for online reviews application.
2. Assign NLP to each word describing in a document. Those documents consist of the extracted task and frequency of each word.

3. Using this data received from the developer have calculated the real time result and developed a documentation to categorize extracted task using machine learning approach.

Our contextual action defines tasks that have been mentioned in documentation. We used open NLP parser which is useful to improve relationships between words. For example, the simple task is "updating page into a document" can be "updated page", "page is updated", "updating page" etc. In this task extraction technique, we used number of methods like task filtering, dependency task, and task modification etc. We also focussed on the extracted task is based on which type of category by calculating the frequency of each task and document. This is useful to increase the performance of the system.

II. RELATED WORK

In this section we review some related works which addresses documentation related issues in task extraction. There are many techniques which are already available so this section gives the review of all these techniques. Such extracting issues are very important in software documents. There are many techniques which are already available so this section gives the review of all these techniques. Also compared the propose approach with these available approaches and defines that how the system provides extraction technique to developers.

Treude, Martin P. Robillard, and Barth_el_emy Dagenais described automatically extracting tasks from software documentation is introduced. The tasks which are extracted from documentation that already have been mentioned. Their proposed system is platform independent, which does not require any machine learning languages. One of the important feature of their work is it enables task navigator through the task extracted which is useful for search queries of user interface. They show that software documentation contains sufficient information to extract development tasks and it is use to reduce an information gap between software documentation and developers. This approach is more helpful to developers for identified search results with task of development. Their work based on different techniques for task extraction like natural language processing, statistical techniques, syntactical analysis of text etc [1].

Samir Gupta, Sana Malik proposed a part of speech parsing includes pos tagger and syntactic parser for source code names. They present a POS tagger and syntactic chucker for source code names that takes into account programmers naming conventions to understand the regular, systematic ways a program element is named. They study the naming conventions used in Object Oriented Programming and identified different grammatical constructions that characterize a large number of program identifiers. This study then informs the design of the POS tagger and chucker. Their evaluation results show a significant improvement in accuracy of POS tagging of identifiers, over the current approaches [3].

S. L. Abebe and P. Tonella**,** defines the approaches of natural language parsing to sentences constructed from the terms that appear in program element identifiers. And also shows parsing can be represented as a dependency tree. They automatically extract an ontology by mapping linguistic entities (nodes and relations between nodes in the dependency tree) to concepts and relations among concepts. This paper also shows that how queries including ontology concepts reduce the search space, when determining the files relevant to the change request. They have extracted domain concepts from program element names by applying NLP and organizing such concepts into ontology. To validate their approach, they have used the concepts in the ontology to (re-formulate) queries used in concept location [5].

Next, we have focused on query expansion work by Haiduc et al. approach based on the query of properties to improve performance of reformulation strategy by automatically recommends given query. Machine learning consists of training set of data which is trained properly with a sample number of queries and relevant results [11]. Yang et al. used the context in which query words are found to extract synonyms, antonyms, abbreviations, and related words for inclusion in the reformulated query [12]. A similar tool for query expansion was described by Hill et al. They extracted noun phrases, verb phrases, and prepositional phrases from method and field declarations. Based on an initial query, their approach returns a hierarchy of phrases and associated method signatures [13]. Finding software based, semantically similar words was also the focus of the work by Howard et al. Their technique mines semantically similar words by leveraging comments and programmer conventions [14]. Again, the main difference to our work is that task extraction suggestions appear after just three typed characters and help the user complete the query rather than reformulate it.

III. PROBLEM STATEMENT

To propose a highly centralized task extraction approach to keep track of the actual usage of the extraction data in the customize portal to meet the text mining, syntactical analysis and natural language processing requirement from the software documentation, which contains all documentation types and also it supports a variety of task extraction methods, categorized each relevant task into different phases of documentation like documentation, development, testing and other etc. taken a proper feedback from the developers.

IV. PROPOSED SYSTEM

The task extracted from software documentation lead to a number of issues which are related to mismatch information to developers. So it is necessary to provide an effective mechanism which monitors all the usage of the task on documentation. There are some techniques which already available as we discuss in literature survey but they have some limitations. So here we are trying to overcome those limitations with the help of applying customize portal framework. All data can be securely extract on documentation is the main purpose of this mechanism. In the existing system, this technique is only applied to particular application but our proposed system apply this framework to document file types which contains all task types, also it also supports a natural language techniques, like text mining for text files, usage

control for extract files. In existing system, they directly extracted task from software documentation but in our idea we are extracted task and then further categorized task into different phases i.e. documentation, development, testing and other then take feedback from developers in the form of comment. The detailed flow of our work is shown in next section.

A. Flow Diagram of Proposed System

In this section, how proposed system will work is elaborated through the following architecture diagram shown in figure 6.1. Basically our whole project works modified and then trying to build a customize application that is to give us best result like accuracy in terms of frequency, categorization of task. This approach can be useful to improve the precision of tasks of software documentation. So that it's become easy to reduce a live space between information which a developer wants and software documentations. Admin uploads document on the server. Whenever admin loads task the machine leaning technique is used. The document can be categorized into different phases and also calculated frequency of each extracted task. Based on that admin add number of tasks which is showed by users. Users views documents and extracted task from admin whichever user wants. Whenever users extracted task at that time admin views all user's data which they have extracted. User can comment about extracted task either positive or negative in the form of feedback. This dissertation is used supervised algorithm because all data trained properly with input and output then it is delivered to the number of users. With the help of text mining we summarized data after applying lucene algorithm for indexing of documentation. And each index entries stored frequency of each word in given document.

The documentation corpus of a project is also pre-monitored so that to get the extraction of development tasks can be done by transforming HTML files into text files which kept information as it is. The documentation of each page contains redundant information like summaries, headers and footers which is repeated and also from the files it discarded. During pre-processing the meta-information kept as it is to shows that HTML header represents a paragraph and code is marked up explicitly. By using stand ford NLP toolkit, the resulting files contains pre-tags which is removed and tokens as well as sentences which is also spited into them.

The main idea is to identify each sentence by the NLP toolkit, the tokens and the grammatical dependencies between tokens are stored into files that contain development tasks or relevant concepts can be explicitly excluded from the analysis. In the case to exclude automatically-generate indexes, release notes, and download instructions. Text Mining usually involves the process of structuring the input text deriving patterns within the structured data, and finally evaluation and interpretation of the data. Text mining takes input as a documentation element where each contains index entry using lucene algorithm. After completion of this process, Natural Language Processing (NLP) is applied. NLP is applied in such a way that the task is to be filtered and structured with HTML files. HLML files are then further transferred into text files based on pre-processing. The extracted tasks are divided into different phases of software engineering (i.e. documentation, development and testing etc.). And finally we calculated the accuracy of the extracted task with frequently asking questions to developers gives a proper feedback.

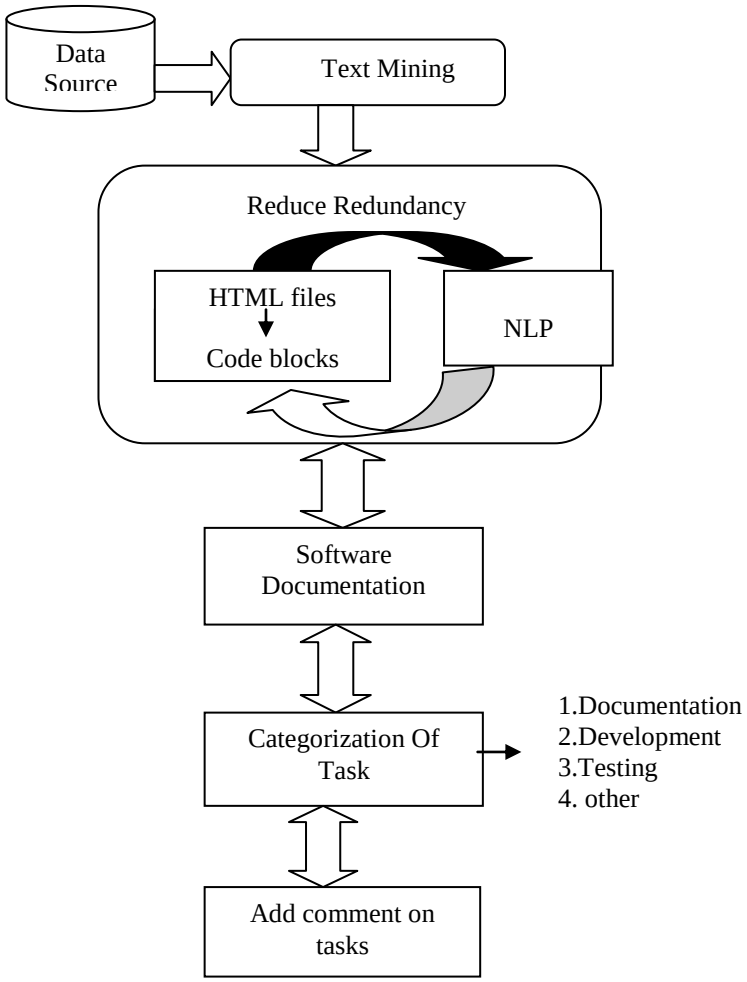

Fig.2. Proposed system architecture

B. Propose Approaches:-

In our research system different approaches are used as follows;-

1. WordNet Library:

The dictionary based approach is WordNet library. We should follow this approach because WordNet library contains a set of words from which to find the way of words.

2. Part of Speech Tagging (POS):-

Part-of-Speech tagging is helpful to mine semantically correlated words from software documentation. The use of part-of-speech (POS) in software engineering based on many text tools which recognize a word as POS and a verbs, nouns, adverbs adjectives as tags etc.

C. Proposed algorithm

In our proposed system, we used lucene algorithm for text mining

I. Natural Language Processing:

The process of interaction between natural languages and computers distributed in artificial intelligence and computer science is called as natural language processing (NLP). It has ability to process the sentences into natural language like English. Following are the steps to execute this technique,

1.Sentiment analysis :

A set of documents usually extract information of subjective to determine divergence regarding definite objects using online reviews. Our application is useful for online reviews to extract relevant task using this sentiment analyzer. It analyzes whole document from large corpus of system and also classify to express whether the given extracted tasks are meaningful to developers or not.

2.Tokenization :

This process that spited sentences into stream tokens such as words. It takes one by one sentence to tokenize each word.

3.Stop words filtering:-

It eliminates stop words from documentation.

II. Lucene indexing algorithm:-

Lucene is one of the most famous algorithms in text mining for indexing and searching documents which is useful to open source Java library. A term consists of pair of string elements which is the basic unit for searching.

Step 1: first choose weighting scheme of text.

Step 2: Apply TF-IDF popular scheme of weighting approach.

Step 3: Then calculated how many times term appears in a document.

Step 4: And also evaluated document frequency which have given term of number of documents.

Step 5:

The equation of finding terms and respective documents are,

$$tf \,.\, idf\,(t, d) = tf(t, d) \,.\, idf(t)$$

$$tf\,(t,d) = \sum_{i \in d}^{id\,j} \{di = t\}$$

idf(t)=

$$\log \left[\frac{|D|}{\sum_{d \in D}^{|D|} |t \in d|} \right]$$

Step 6: End

C. Experimental Results

In our approach, data admin create and upload documentation files on the server. Then admin add relevant task and according to categorize then assign to respective users. Then users search and click on extracted task and give comment like positive or negative about extracted task Admin view all comment about task which has been given by user and also add comment regarding of that task. So whenever next user view task then user will get all information of extracted task. Our dissertation work is done which is shown in this section. Our results shows that the extraction of task and taking feedback from developer performs better results in terms of accuracy through natural language processing.

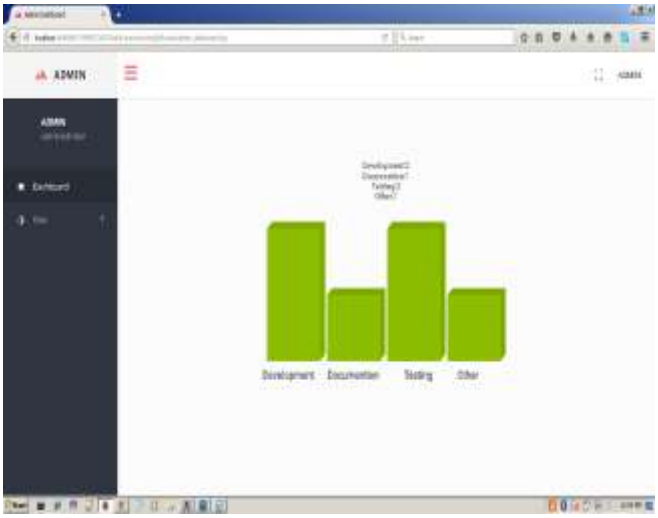

Fig.ure 3. Categorization of tasks

In above result, this is dashboard of admin of customize portal. The system is categorized relevant tasks into four phases of software documentation i.e development, documentation, testing and other etc. using machine leaning techniques. And also calculated frequency of each task mapped it in a bar graph and admin is assigned it according to user needs. Admin has authority that which task is given to which user. With the help of this, user will understand how many tasks are there in our system. We can create number of tasks of software documentation in our system.

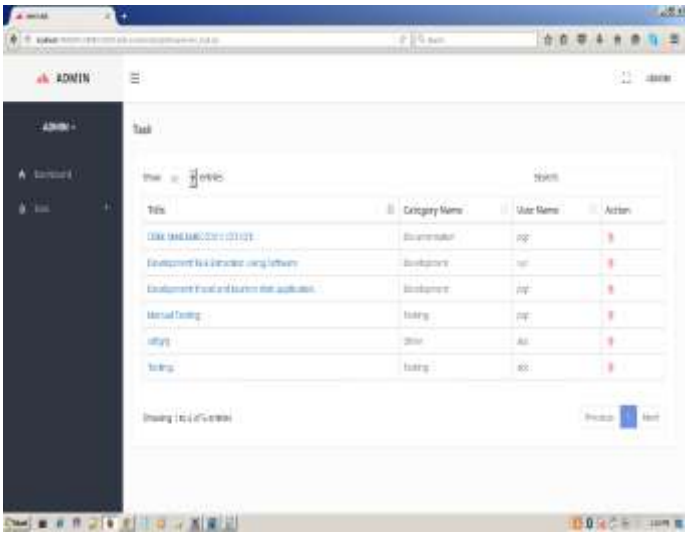

Fig.ure 4. Description of tasks and user information

In fig.4 result shows that a detail of all tasks, category of task, user name and action based on that task is deleted or not. It also describes task details like title of task by using this it gives us which kind of task is mentioned.

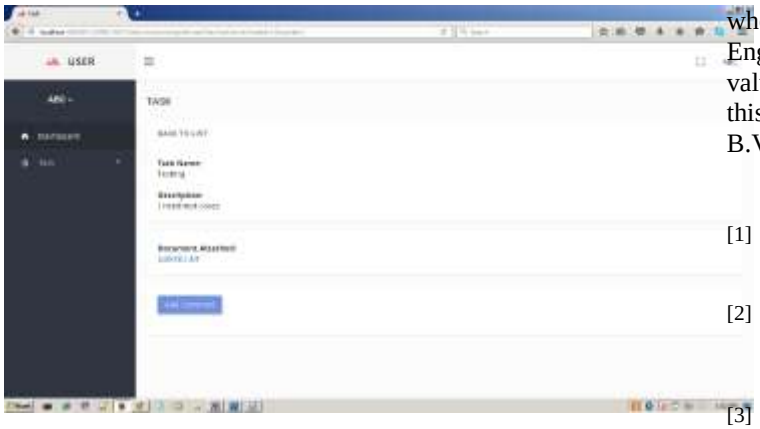

Figure 5. User add comment about extracted task

This system contains user dashboard of customize portal which contains extracted task information along with document attachment. User can add comment on extracted task either positive or negative. Here we apply sentiment analysis for classification of comments. This result contains task name, description of task, document file and comment. The comment is in the form of user feedback for analysis of result and also gives better performance.

V. CONCLUSION

In this paper, new approach for automatically extracting with natural language processing technique is introduced. This approach allows the developers of the data to not only extract his data but also categorized relevant task into different software phases like documentation, development, testing and other etc. In this concept, NLP is used to get the required data and also used text mining, machine learning, sentiment analysis method and some more algorithms to calculate the results. The result shows that tasks can be extracted from customize portal and also reduced a live space between information which developers want and software documentation also developers can add comment about extracted tasks.

ACKNOWLEDGMENT

This is to acknowledge and thank all the individuals who played defining role in shaping this paper. Without their constant support, guidance and assistance this paper would not have been completed. Without their Coordination, guidance and reviewing this task could not be completed alone. I avail this opportunity to express my deep sense of gratitude and whole hearted thanks to my guide and head of computer Engineering. Dept. Prof. Dr. D. M. Thakore for giving his valuable guidance, inspiration and encouragement to embark this paper and our Honble principal Dr.Anand Bhalerao at B.V.D.U College of Engineering, Pune (BVDUCOE).

REFERENCES

[1] Christoph Treude, Martin P. Robillard, and Barth_el_emy Dagenais ,"Extracting Development Task To Navigate Software Documentation" in Proc, IEEE Soft,Vol.41 No.6,2015,pp,565-581

[2] S. Gupta, S. Malik, L. Pollock, and K. Vijay-Shanker, "Part-ofspeech tagging of program identifiers for improved text-based software engineering tools," in Proc. 21st IEEE Int. Conf. Program Comprehension, 2013, pp. 3–12.

[3] M. Barouni-Ebrahimi and A. A. Ghorbani, "On query completion in web search engines based on query stream mining," in Proc. IEEE/WIC/ACM Int. Conf. Web Intell., 2007, pp. 317–320.

[4] P. Mika, E. Meij, and H. Zaragoza, "Investigating the semantic gap through query log analysis," in Proc. 8th Int. Semantic Web Conf., 2009, pp. 441–455.

[5] S.L.Abebe and P.Tonella,"Natural language parsing of program element names for concept extraction," in Proc. 18 th IEEE Int. Conf. Program Comprehension, 2010, pp. 156–159.

[6] C. Treude and M.-A. Storey, "Effective communication of software development knowledge through community portals," in Proc. 8th Joint Meet. Eur. Soft. Eng. Conf. ACM SIGSOFT Symp. Found. Soft. Eng., 2011, pp. 91–101.

[7] T. C. Lethbridge, J. Singer, and A. Forward, "How software engineers use documentation: The state of the practice," IEEE Soft., vol. 20, no. 6, pp. 35–39, Nov./Dec. 2003.

[8] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky, "The Stanford Core NLP natural language processing toolkit," in Proc. 52nd Annu. Meet. Assoc. Computat. Linguistics: Syst. Demonstrations, 2014, pp. 55–60.

[9] G. Sridhara, E. Hill, L. Pollock, and K. Vijay-Shanker, "Identifying word relations in software: A comparative study of semantic similarity tools," in Proc. 16th IEEE Int. Conf. Program Comprehension, 2008, pp. 123–132.

[10] H. Zhong, L. Zhang, T. Xie, and H. Mei, "Inferring resource specifications From natural language API documentation," in Proc. 24th IEEE/ACM Int. Conf. Automated Soft. Eng., 2011, pp. 307–318.

[11] S. Haiduc, G. Bavota, A. Marcus, R. Oliveto, A. De Lucia, and T. Menzies, "Automatic query reformulations for text retrieval in software engineering," in Proc. 35th Int. Conf. Soft. Eng., 2013, pp. 842–851.

[12] J. Yang and L. Tan, "Inferring semantically related words from software context," in Proc. 9th Working Conf. Min. Softw. Repositories, 2012, pp. 161–170.

[13] E. Hill, L. Pollock, and K. Vijay-Shanker, "Automatically capturing source code context of NL-queries for software maintenance and reuse," in Proc. 31st Int. Conf. Soft. Eng., 2009, pp. 232–242.

[14] M. J. Howard, S. Gupta, L. Pollock, and K. Vijay-Shanker, "Automatically mining software-based, semantically-similar words from comment-code mappings," in Proc. 10th Working Conf. Min. Softw. Repositories, 2013, pp. 377–386.