International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 06 Issue: 02 | Feb 2019 www.irjet.net p-ISSN: 2395-0072

A Novel Approch Automatically Categorizing Software Technologies

Khushang Mehta[1], Prof. Kore Kunal Sidramappa[2]

[1,2]*Sharadchandra Pawar College of Engineering, Otur, Pune, India*

Abstract—Software development is increasingly based on reusable components in the form of frames and libraries, as well as programming languages and tools to use them. Informal language and the absence of a standard taxonomy for software technologies make it difficult to reliably analyze technologi-cal trends in discussion forums and other online sites. The system proposes an automatic approach called Witt for the categorization of software technology. Witt takes as input a sentence that describes a technology or a software concept and returns a general category that describes it (for example, an integrated development environment), along with attributes that qualify it even more. By extension, the approach allows the dynamic creation of lists of all technologies of a given type.The system contribute Levenshtein distance algorithm to compare similarities between two stings.It work on character distances of two strings.With this algorithm it is possible to categorize the data from large data.

Index Terms—hypernym, Lexicography, NLP, Software technologies.

I. INTRODUCTION

Now days the Software development is increasingly based on reusable components platform in the form of frames and libraries, as well as programming languages and tools to use them. Taken together, these software technologies form a massive and rapidly growing catalog of constituent elements for systems that becomes difficult to monitor through dis-cussion channels. The list of all technologies of a certain type or their popularity in relation to this type. Questions like ”what is the most popular web application framework?” They are important for many organizations, for example, to decide which development tool to adopt at the beginning of a project or for which technology to develop a driver. The answers to these questions are routinely proposed without any supporting data, but it is difficult to find valid empirical surveys. To move to a rationalized, evidence-based approach to monitor the use of software technologies, we must be able to automatically classify and group the nominated mentions of software technologies.

A. MOTIVATION

An important step towards understanding the terminology of the machine is the discovery of hypernyms, that is, the discovery of the more general concept in an is-a relationship (for example, AngularJS is a web application framework), which has led to the development of many tools hypernyms automated extraction. Unfortunately, the discovery of correct hypernyms is not efficient to support the detection and moni-toring of comparable software techniques. For example, the cross-platform commercial IDE for PHP is a hyper valid for Php Storm, but the expression is too specific to make a useful category of technologies. The categorization of software technologies is a much more complex problem that requires greater abstraction and normalization.

B. OBJECTIVE

To extract general categories and related attributes for the hypernyms.To get data for user entered query.

II. REVIEW OF LITERATURE

1. Present natural language processing to extract important concepts from identifiers defined in source code, aggregating them into a WordNet-like structure that includes their hypernyms relation [1].

2. Present One of the main limitations of WordNet for software engineering applications is the lack of support for specialized terminology. A number of projects have focused on the design of lexical databases that include a word similarity relationship. This relationship can be calculated from the co-occurrences in the context of a forum publication. [2]

3. Presented in a study on the use of labels in a book management system, Treude and Storey discovered that software developers had developed implicit and explicit mechanisms for managing label vocabularies. [3].

4. Current system that calculates the textual similarity based on the similarity of grams between the first paragraph of the section of the selected article and the extract of the label. This metric calculates the proportion of common token q sequences between two strings. The result is a score between 0 (completely different) and 1 (exact copies). This system chose the commonly used value q = 2 for this metric, with words such as token. This system derived each word using Porter Stemmer and did not consider the envelope of the letter[4].

5. WordNet is a lexical database that contains, among other information, hyperimone and hyperimone relationships. The database was created manually and is considered a gold standard in many language applications. With WordNet, the system first retrieved all the words that matched the label. This system eliminated a result if its brightness did not

contain at least one of the programming axes. For all the remaining results, this system analyzed the words listed as hypernotic in the result. This system considers all those hypermitos for the evaluation. [5].

6. Present system that mine data from millions of questions from the QA site Stack Overflow, and using a discriminative model approach, this system automatically suggest question tags to help a questioner choose appropriate tags for eliciting a response[6].

7. Present a new algorithm to learn hypernyms relationships (is-a) from the text, a key problem in machine learning for the understanding of natural language. This method generalizes the previous work that was based on synthetic-lexical patterns constructed by hand, introducing a general formalization of the space of patterns based on paths of syntactic dependence[7].

8. Propose a semantic graph to model semantic correlations between labels and words in software descriptions. Then, according to the graph, this system designed an effective algorithm to recommend labels for the software. With full experiments on large scale open source software data sets that compare them to different typical related jobs, this system demonstrate the effectiveness and efficiency of our method to recommend appropriate labels[8].

9. This paper proposes a simple and general technique to automatically deduce semantically related words in the software using the context of the words in the comments and in the code. In addition, the Present system offers a classification algorithm in the results of rPair and studies pairs of crossed projects in two sets of software with similar functionality, that is, multimedia browsers and operating systems.

10. This paper addresses one of these main obstacles, namely, the lack of adequate mechanisms of programmatic access to the knowledge stored in these large semantic knowledge bases. The current system features two application programming interfaces for Wikipedia that are specifically designed to extract lexical semantic information enriched in knowledge bases and provide efficient and structured access to available knowledge[10]

III. SYSTEM ARCHITECTURE/ SYSTEM OVERVIEW

In the proposed system, our approach takes as input a term to categorize. As a vocabulary for the software technology system, they have data of all the methodologies, so the system gets the data labels. According to the label, they will obtain all the data coming from a different technology. Apply NLP and Levenshtein distance algorithm. Then hypernyms will find like final step of the proposed system contains of transforming the hypernyms into a set of categories, possibly with some attributes. This system designed categories to represent general hypernyms, with a focus on coverage: commercial ide for php is a better (more precise) hypernyms than ide, but the latter is a better category (higher coverage). The attributes are meant to provide a flexible way to express the information lost when transforming a hypernyms into a category. They represent typical variants of the category, but would not constitute valid hypernyms on their own. To transform a hypernyms into category with attributes, this system start by removing all non-informative phrases like name of and type of this system also transform phrases indicating a collection, e.g., set of, into the attribute collection of, and remove it from the hypernyms. This system constructed a small list of such phrases based on our development set. If two or more occurrences of the word of or of the word for remain in the hypernyms, this system do not parse the hypernyms, as its structure is possibly too complex for our simple heuristics.The system contribute Levenshtein distance algorithm to compare similarities between two stings.In information theory, and computer science, the Levenshtein distance is a string metric for calculating the difference between two sequences. Informally, the Levenshtein distance between two words is the minimum number of single-character edits (insertions, deletions or substitutions) required to change one word into the other. The application will work on computer science related information.

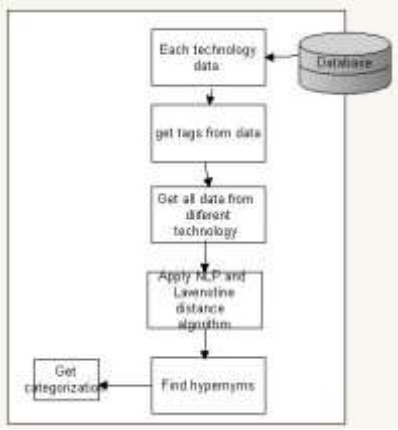

Fig. 1. Proposed System Architecture

A. Algorithms

1) NLP:

Natural language processing (NLP) is a subfield of computer science, information engineering and artificial intelligence that deals with the interactions between computers and (natural) human languages, especially how to program computers to process and analyze large quantities of data. In natural language.

2) Similarity score:

Calculate the string similar based on the similarity of the grams Q between the first paragraph of the section of the selected article and the extract of the label. The similarity is calculated for the first line of both texts, then the first two sentences, the first three, etc., till one of the inputs

parameter runs out of sentences. The best similarity score is keep representative of the overall similarity between the two inputs parameter.

3) Levenshtein distance algorithm:

The Levenshtein distance is a string metric to measure the difference between two sequences. Informally, the Levenshtein distance between two words is the mini-mum number of single-character edits (i.e. insertions, deletions or substitutions) required to change one word into the other.

The Levenshtein algorithm: calculates the least number of edit operations that are necessary to modify one string to obtain another string. The most common way of calculating this is by the dynamic programming approach. In proposed system Present system using this to match user entered question with available question in database.

Input. : Get user entered question. Working:

Step1. Select user entered query

Step 2:. Select all data from available database

Step3. Pass the distance to match query question with available data.

System will check question with according to entered query with available data. word by word with available answer.

Step4: One by one query will gets by visiting each data to specified distance.

Output: Get matched similar data.

B. Mathematical Model

This will be used to calculate accuracy in proposed system.It categorize query and result data from system. In the field of information retrieval, precision is the fraction of retrieved documents that are relevant to the query:

$$\text{precision} = \frac{\text{relevantdocument} \backslash \text{retrieveddocuments}}{\text{retrieveddocuments}}$$

In information retrieval, recall is the fraction of the relevant documents that are successfully retrieved.

$$\text{recall} = \frac{\text{relevantdocument} \backslash \text{retrieveddocuments}}{\text{relevantdocuments}}$$

In binary classification, recall is called sensitivity. It can be viewed as the probability that a relevant document is retrieved by the query.

C. HARDWARE AND SOFTWARE REQUIREMENTS

Hardware Requirements

1) Processor - Intel i5 core
2) Speed - 1.1 GHz
3) RAM - 2GB
4) Hard Disk - 40 GB
5) Key Board - Standard Windows Keyboard
6) Mouse - Two or Three Button Mouse
7) Monitor - SVGA

Software Requirements

1) Operating System - XP, Windows7/8/10
2) Coding language - Java, MVC, JSP, HTML, CSS etc
3) Software - JDK1.7
4) Tool - Eclipse Luna
5) Server - Apache Tomcat 7.0
6) Database - MySQL 5.0

IV. SYSTEM ANALYSIS AND RESULT

Experimental setup Table 1-The proposed system string categorize, it gives efficient time to categorize document according to entered string.Fig.2-Graph showed a pictorial rep-resentation of No.of matched document time. X-Axis contains no.of document and y-axis time to match query.Graph shows in proposed system how search time varies with respect to the number of documents. in our implementation, search time depends not only on the number of documents returned, but also on the number of documents in which the query to be categorize are present.

TABLE I

EXECUTION TIME FOR CATEGORIZE THE ENTERED QUERY IN NO.OF DOCUMENTS.

Index	No.of documents	Query	Time to catego-rize(ms)
1	1000	Query1	101
2	2000	Query2	140
3	3000	Query3	190

V. CONCLUSION

In this paper, System proposed a novel a domain-specific technique to automatically produce an attributed category structure describing an input phrase assumed to be a software technology. Here found that after trans-forming hypernyms into more abstract categories. Our approach takes

as input a term to categorize software. It uses NLP and Levenshtein distance algorithm.

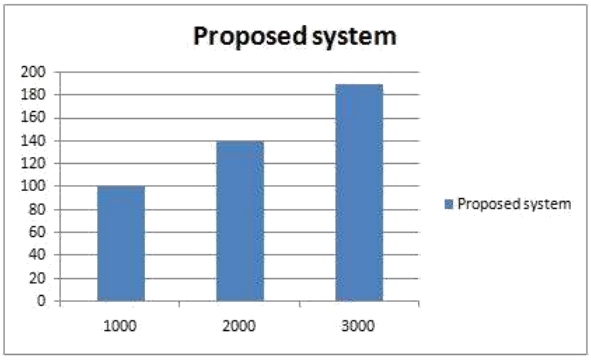

Fig. 2. Categorize time for no.of documents.

ACKNOWLEDGMENT

The authors would like to thank the researchers as well as publishers for making their resources available and teachers for their guidance. We are thankful to the authorities of Savitribai Phule University of Pune and concern members of cPGCON2019 conference, organized by, for their constant guidelines and support. We are also thankful to the reviewer for their valuable suggestions. We also thank the college authorities for providing the required infrastructure and support. Finally, we would like to extend a heartfelt gratitude to friends and family members.

REFERENCES

[1] J.-R. Falleri, M. Huchard, M. Lafourcade, and M. Dao, Automatic extraction of a WordNet-like identifier network from software, in 18th IEEE International Conference on Program Comprehension (ICPC), 2010, pp. 413.

[2] SEWordSim: Software-specific word similarity database, in Companion Proceedings of the 36th International Conference on Software Engineer-ing, 2014

[3] C. Treude and M.-A. Storey, Work item tagging: Communicating concerns in collaborative software development, IEEE Transactions on Software Engineering, vol. 38, no. 1, 2012.

[4] M. F. Porter, An algorithm for suffix stripping, Program, vol. 14, no. 3, 1980.

[5] G. A. Miller, R. Beckwith, D. Gross, and K. J. Miller, Introduction to Wordnet: An on-line lexical database, International Journal of Lexicog-raphy, vol. 3, no. 4, 1990.

[6] A. K. Saha, R. K. Saha, A discriminative model approach for suggesting tags automatically for Stack Overflow questions, in Proceedings of the 10th Working Conference on Mining Software Repositories, 2013, .

[7] R. Snow, D. Jurafsky, and A. Y. Ng, Learning Syntactic Patterns for Automatic Hypernym Discovery, in Proceedings of the 18th Annual Conference on Neural Information Processing Systems, 2004.

[8] T. Wang, H. Wang, G. Yin, X. Li, and P. Zou, Tag recommendation for open source software, Frontiers of Computer Science, vol. 8, no. 1, , 2014.

[9] J. Yang and L. Tan, SWordNet: Inferring semantically related words from software context, Empirical Software Engineering, pp. 131, 2013.

[10] T. Zesch, C. Mller, Extracting lexical semantic knowledge from wikipedia and wiktionary, in Proceedings of the Conference on Language Resources and Evaluation, electronic proceedings, 2008.