© July 2019 | IJIRT | Volume 6 Issue 2 | ISSN: 2349-6002

Automatically Categorizing Software Technologies

Dinesh Hemant Bhere[1], Prof.K.S.Kore[2], Prof Monika Rokde[3]
[1,2,3]*Sharadchandra Pawar College of Engineering, Otur, Pune*

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 technological 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- NLP, data mining, Lexicography, Hypernym

I. INTRODUCTION

Software development increasingly relies on reusable components in the forms of frameworks and libraries, and the programming languages and tools to use them. Considered together, these software technologies form a massive and rapidly- growing catalog of building blocks for systems that becomes difficult to monitor across discussion channels. The unstructured data, informal nomenclature, and folksonomies used on social media forums make it difficult to reliably determine, for example, the list of all technologies of a certain type, or their popularity relative to this type. Questions such as what is the most popular web application framework? are important to many organizations, for example to decide which development tool to adopt at the start of a project, or which technology to develop a driver for. Answers to these questions are routinely proposed without any kind of supporting data, but sound empirical surveys are hard to find. To move towards a streamlined, evidence-based approach to monitoring the use of software technologies, Present system need to be able to automatically classify and group named mentions of software technologies. An important step toward the machine understanding of terminology is hypernym discovery, i.e., the discovery of the more general concept in a is a relationship (e.g., Angular JS is a web application framework), which led to the development of many automated hypernym extraction tools. Unfortunately, discovering valid hypernyms is not sufficient to support the detection and monitoring of comparable software technologies. For example, commercial cross-platform IDE for PHPis a valid hypernym for PhpStorm, but the expression is too specific to constitute a useful category of technologies. Categorizing software technologies is a much more complex problem that requires additional abstraction and normalization. To address this issue, Present system propose an automated approach for the categorization of software technologies. Our approach, called Witt, for What Is This Technology, takes as in-put a term such as Php Storm and returns a general category that describes it (e.g., integrated development environment),along with attributes that further qualify it (comercial,php,etc.). Our approach involves three automated phases that each address a major technical challenge.

II. SYSTEM ARCHITECTURE

In proposed system this approach takes as input a term to categorize. As a vocabulary for software technologies system have data of all technology then system gets tag of data. According to tag all data from different technology will get. Apply NLP and Levenshtein distance algorithm. Then hypernyms will find like final step of the approach consists of transforming the hypernyms into a set of categories, possibly with some attributes. Present system designed categories to represent general hyponyms, 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 hyponyms on their own. To transform a hyponym into category with attributes, Present system start by removing all non-informative phrases like name of and type of Present system also transform phrases indicating a collection, e.g., set of, into the attribute collection of, and remove it from the hypernym. Present system constructed a small list of such phrases based on this development set. If two or more occurrences of the word of or of the word for remain in the hypernym, Present system do not parse the hypernym, as its structure is possibly too complex for this simple heuristics.

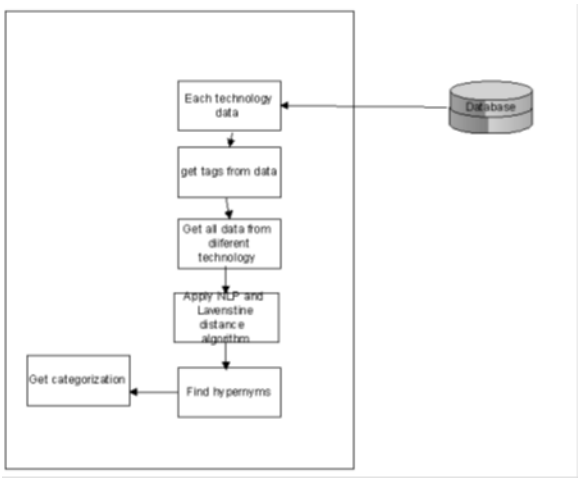

Each technology data
Database
get tags from data
Get all data from different technology
Apply NLP and Lavenstine distance algorithm
Get categorization
Find hypernyms

Fig.1: System Architecture

III. SYSTEM MODEL DESIGN PROCEDURE

The implementation modules of the system is shown in the fig.2

A. Explanation:

1) Get tags-

The purpose of the get tags is to get all tag from database to categorize the software data.

2) Get all data of tag-

This is the main part of used to get all data of mined tag from different domain.

3) Apply NLP and Levenshtein distance algorithm-

The main function of this module is to apply NLP that is pre-processing of data. Then for similarity Levenshtein distance algorithm is performed.

4) Categorize software-

This module is used to execute that finally categorize the software

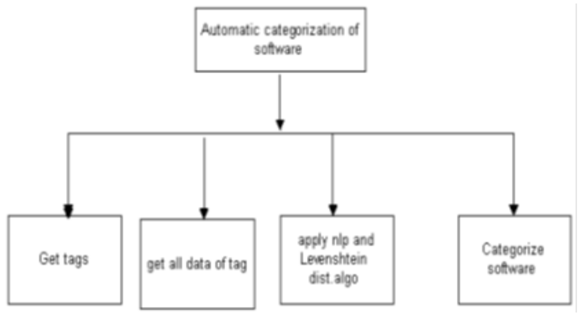

Automatic categorization of software
Get tags
get all data of tag
apply nlp and Levenshtein dist.algo
Categorize software

Fig.2: Module Design

B. Algorithm

Algorithm 1: Natural Language processing

1. Lexical Analysis: It involves identifying and analyzing the structure of words. Lexicon of a language means the collection of words and phrases in a language. Lexical analysis is dividing the whole chunk of txt into paragraphs, sentences, and words.
2. Syntactic Analysis (Parsing): It involves analysis of words in the sentence for grammar and arranging words in a; manner that shows the relationship among the words. The sentence such as The school goes to boy is rejected by English syntactic analyzer. 3. Semantic Analysis: It draws the exact meaning or the dictionary meaning from the text. The text is checked for meaningfulness. It is done by mapping syntactic structures and objects in the task domain. The semantic analyzer disregards sentence such as hot ice-cream.
3. Discourse Integration The meaning of any sentence depends upon the meaning of the sentence just before it. In addition, it also brings about the meaning of immediately succeeding sentence.
4. Pragmatic Analysis: During this, what was said is re-interpreted on what it actually meant. It involves deriving those aspects of language which require real world knowledge.

Algorithm 2: Levenshtein distance algorithm.

The Levenshtein algorithm (also called Edit-Distance) 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.

IV. RESULTS

Experimental setup Table 1-The proposed system string categorize, it gives efficient time to categorize document according to entered string. Fig.3 Graph showed a pictorial representation of No. of tags of each language. X-Axis contains no. of languages and y-axis no of tags.

Table 1: No. of tags of each language

Index	Query	No. of tags(ms)
1	Java	25
2	Cpp	50
3	C	35
4	Sql	45

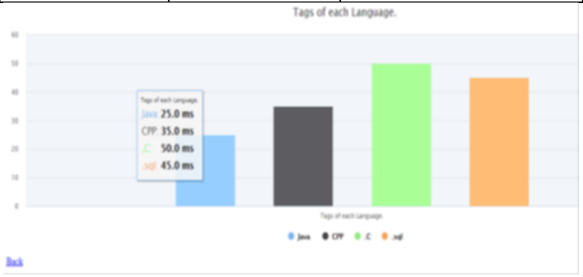


Fig.3: No.of tags of Language

A NOVEL APPROCH AUTOMATICALLY CATEGORIZING SOFTWARE TECHNOLOGIES
User
Admin

Fig.4: Home.jsp (This is home page of project)

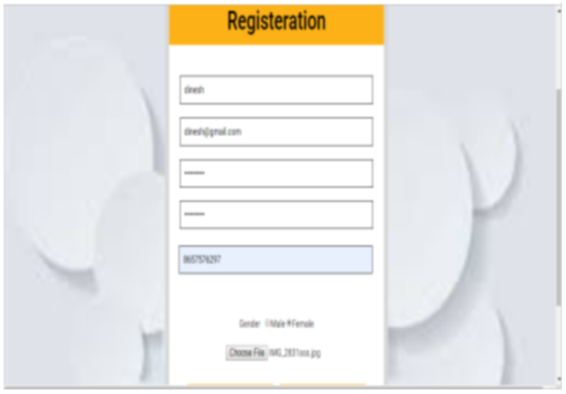


Fig.6: Registration.jsp (this is registration page where user enters his details)

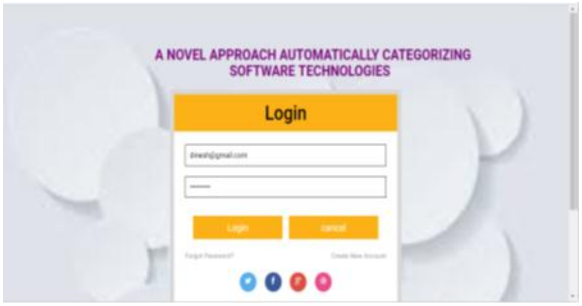


Fig.7: Login.jsp (this is login page to access the system)

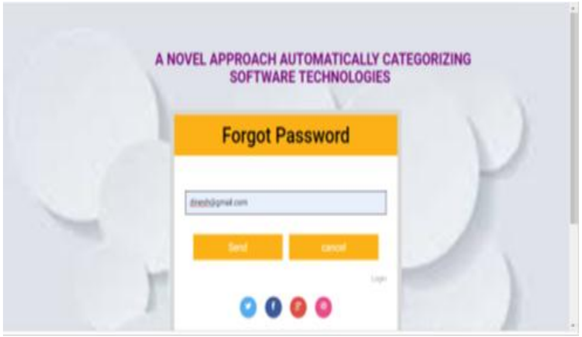
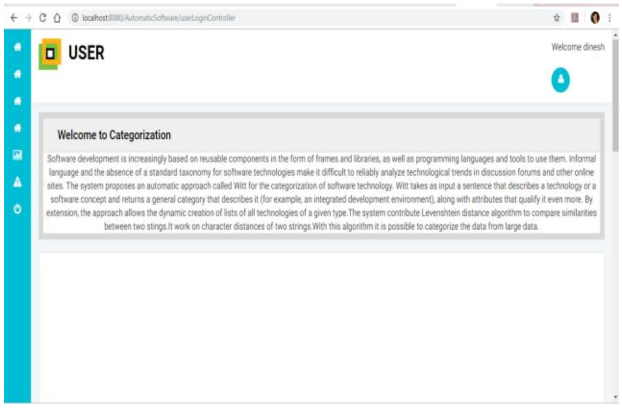

A NOVEL APPROACH AUTOMATICALLY CATEGORIZING SOFTWARE TECHNOLOGIES
Forgot Password
dinesh@gmail.com
Send
cancel
Login

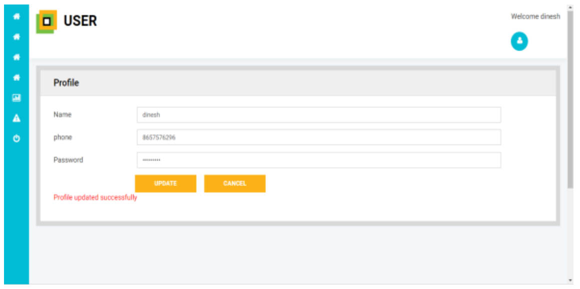
Fig. 8: Forgotpassword.jsp. (this is forgot password page to get password of user account)



Welcome to Categorization

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 technological 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.

Fig. 9: home.jsp (this is home page after successfully login of user)

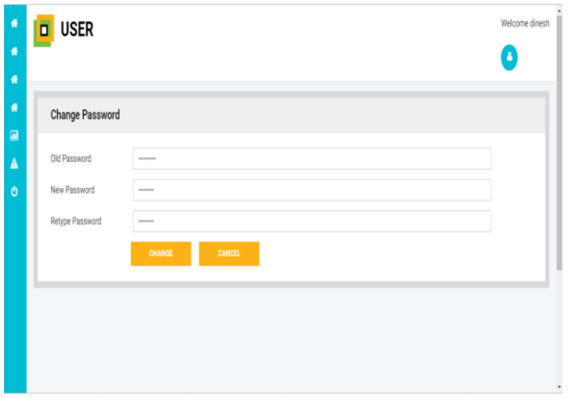


Profile

Name: dinesh
phone: 8657576296
Password: ••••••••

Profile updated successfully

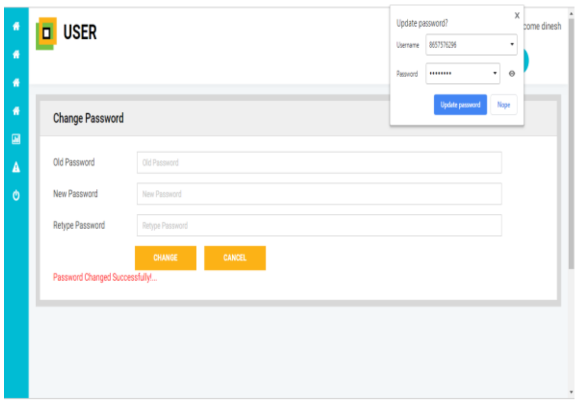
Fig. 10: Profile.jsp (this is profile page of user. User can update his profile)



Change Password

Old Password: ••••••
New Password: ••••••
Retype Password: ••••••

Fig. 11: Change password.jsp (this is change password page here user can change password)



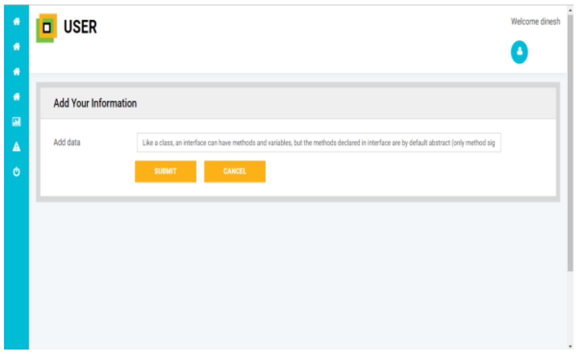
Username: 8657576296
Password: ••••••••

Change Password

Old Password: Old Password
New Password: New Password
Retype Password: Retype Password

Password Changed Successfully!...

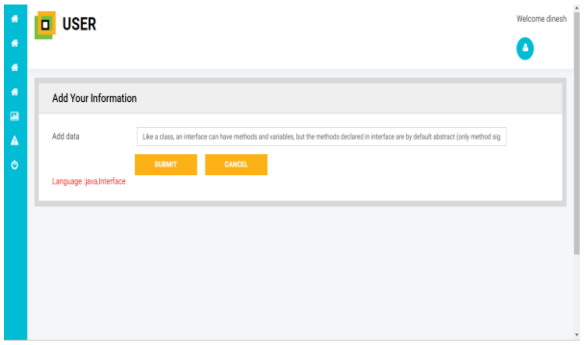
Fig. 12: Passwordchanges.jsp (in this page user changed his password successfully)



Add Your Information

Add data: Like a class, an interface can have methods and variables, but the methods declared in interface are by default abstract (only method sig

Fig. 13: Information to Categorize (on this page user enteres information to categorize)

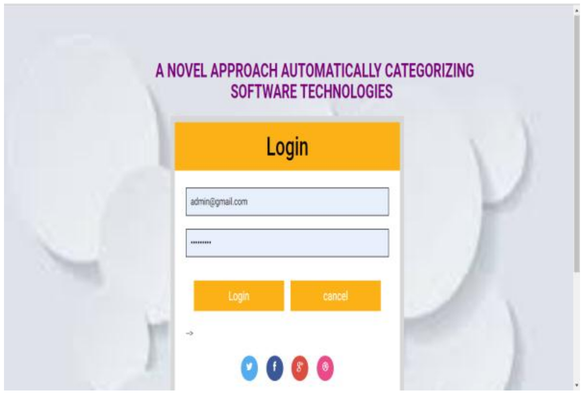


Add Your Information

Add data: Like a class, an interface can have methods and variables, but the methods declared in interface are by default abstract (only method sig

Language :java,Interface

Fig. 14: CategorizedInformation.jsp (On this form user gets category of information)


A NOVEL APPROACH AUTOMATICALLY CATEGORIZING SOFTWARE TECHNOLOGIES
Login
Login
cancel

admin@gmail.com
•••••••••

Fig. 15: Admin Login.jsp (here admin login to system)

V. APPLICATION

1. Data clustering: Clustering is a Machine Learning technique that involves the grouping of data points. Given a set of data points, we can use a clustering algorithm to classify each data point into a specific group. In theory, data points that are in the same group should have similar properties and/or features, while data points in different groups should have highly dissimilar properties and/or features. Clustering is a method of unsupervised learning and is a common technique for statistical data analysis used in many fields.
2. Search engine application: A search-based application is generally taken to mean an application that uses a search engine index at its core, rather than a database or other structure. Search engines are extremely good at slicing and dicing information on-the-y. Search-based applications exploit this capability. Search Technologies designs and builds search-based applications using the leading search commercial products and open sources alternatives

VI. CONCLUSION

In this paper, present system have 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 transforming hypernyms into more abstract categories. this approach takes as input a term to categorize software. It uses NLP and Levenshtein distance algorithm. The proposed scheme have the scope in application in future, such as Business, Educational Systems like school, college, etc.

REFERENCES

[1] J.-R. Falleri, M. Huchard, M. Lafourcade, C. Nebut, V. Prince, and M. Dao, Automatic extraction of a WordNet-like identifer network from software, in 18th IEEE International Conference on Program Comprehension (ICPC), 2010, pp. 4-13.

[2] SEWordSim: Software-speci_c word similarity database, in Companion Proceedings of the 36th International Conference on Software Engineering, 2014, pp.568-571.

[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, pp. 19-34, 2012.

[4] M. F. Porter, An algorithm for suffix stripping, Program, vol. 14, no. 3, pp. 130-137, 1980.

[5] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller, Introduction to Wordnet: An on-line lexical database, International Journal of Lexicography, vol. 3, no. 4, pp. 235-244, 1990.

[6] A. K. Saha, R. K. Saha, and K. A. Schneider, A discriminative model approach for suggesting tags automatically for Stack Overflow questions, in Proceedings of the 10th Working Conference on Mining Software Repositories, 2013, pp. 73^a76.

[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, C. X. Ling, X. Li, and P. Zou, Tag recommendation for open source software, Frontiers of Computer Science, vol. 8, no. 1, pp. 69-82, 2014.

[9] J. Yang and L. Tan, SWordNet: Inferring semantically related words from software context, Empirical Software Engineering, pp. 1^a31, 2013.

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

[11] D. Lo, L. Jiang, and F. Thung, Detecting similar applications with collaborative tagging, in Proceedings of the International Conference on Software Maintenance, 2012, pp. 600-603.

[12] C. D. Manning and H. SchIutze, Foundations of statistical natural language processing. MIT press, 1999

[13] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. Mc-losky, The stanford CoreNLP natural language processing toolkit, in Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, 2014, pp. 550-60.

[14] P. N. Mendes, M. Jakob, A. Garc Silva, and C. Bizer, DBpedia Spotlight: Shedding light on the web of documents, in Proceedings of the 7th International Conference on Semantic Systems, 2011, pp. 18.

[15] R. Mihalcea and A. Csomai, Wikify! Linking documents to encyclopedic knowledge, in Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management, 2007, pp. 233242.

[16] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller, Introduction to Wordnet An online lexical database, International Journal of Lexicography, vol. 3, no. 4, pp. 2350-244, 1990.

[17] D. Milne and I. H. Witten, Learning to link with Wikipedia, in Proceedings of the 17th ACM Conference on Information and Knowledge Management, 2008, pp. 509-518.

[18] K. Nakayama, T. Hara, and S. Nishio, Wikipedia link structure and text mining for semantic relation extraction. in Proceedings of the

Workshop on Semantic Search, 5th European Semantic Web Conference, 2008, pp. 59-73.
[19] J. Nonnen, D. Speicher, and P. Imho, Locating the meaning of terms in source code: Research on term introduction, in Proceedings of the 18th Working Conference on Reverse Engineering,2011, pp. 99-108.
[20] M. F. Porter, An algorithm for suffix stripping, Program, vol. 14, no. 3, pp. 130-137, 1980.
[21] A. Ritter, S. Soderland, and O. Etzioni, What is this, anyway: Automatic hypernym discovery, in Proceedings of the AAAI Spring Symposium: Learning by Reading and Learning to Read, 2009, pp. 88-93.
[22] A. K. Saha, R. K. Saha, and K. A. Schneider, A discriminativ emodel approach for suggesting tags automatically for Stack Overflow questions, in Proceedings of the 10th Working Conference on Mining Software Repositories, 2013, pp. 73-76.
[23] J. Seitner, C. Bizer, K. Eckert, S. Faralli, R. Meusel, H. Paulheim,and S. P. Ponzetto, A large database of hypernymy relations extracted from the web,^a in Proceedings of the 10th edition of the Language Resources and Evaluation Conference, 2016, pp. 360-367.
[24] R. Snow, D. Jurafsky, and A. Y. Ng, Learning Syntactic Patterns for Automatic Hypernym Discovery, in Proceedings of the 18thAnnual Conference on Neural Information Processing Systems, 2004.
[25] C. Stanley and M. D. Byrne, Predicting tags for StackOverowposts, in Proceedings of the 12th International Conference on Cognitive Modelling, 2013, pp. 414-419.