A NOVEL COGNITIVE AGENT SYSTEM TO RETRIEVE RELEVANT CODE COMPONENTS FROM A REPOSITORY

Guide: **Dr. K. JAYA RAJAN**

Asst.Professor, Department of CSE, **MALLA REDDY ENGINEERING COLLEGE FOR WOMEN**, TELANGANA, India

M.SRINIJA REDDY[1], P.SAMSHRITHA[2],N.DEEKSHANA[3],M.SREEJA[4]

B. Tech Final Year, Department of CSE, **MALLA REDDY ENGINEERING COLLEGE FOR WOMEN**

TELANGANA, India

ABSTRACT

Despite its well-recognized benefits, software reuse has not met its expected success due to technical, cognitive, and social difficulties. We have systematically analyzed the reuse problem (especially the cognitive and social difficulties faced by software developers who reuse) from a multidimensional perspective, drawing on our long-term research on information retrieval, human-computer interaction, and knowledge-based systems. Based on this analysis, we propose the concept of reuse-conducive development environments, which encourage and enable software developers to reuse through the smooth integration of reuse repository systems and development environments.

Index terms: Cognitive, Social Difficulties, Systematically, Reuse-Conducive.

1.INTRODUCTION

The growth of code component reusability had increased with most of the developers or end users majorly browse for the required code components in the internet, as it is providing many open source software code components. The user generally enters query in natural language and get plenty of results among which the relevancy of required code component is less, as it contains huge amount of noisy data than relevant data. To overcome this problem the authors introduced a cognitive agent system to retrieve most relevant code component from the repository. As per the work done to implement the concept the authors made use of a latest approach called Code2Vec. This is a neural embedding process of converting the code components into numerical representation called vectors. According to Piyush Arora etal[1], The conversion of code component to vectors can be done in three ways - one is general code as vectors in which the spaces or new lines and stop words are eliminated using tokenizer, another one is tokenization in which it provides the lexical scanner for the code components to convert into vectors, and last one is AST(abstract structure tree) in which the code components are separated depending on their relationship and represented in the form of a tree. In this work natural language processing is used to perform the retrieval. The ode2vec is mainly used to predict the method names. With an idea of fetching the ethod or code snippet of a particular method, it was decided to implement Code2vec concept as it was proven as effective code embedding for predicting the methods. As per Hong jin kang[2], they proposed token embedding's by code2vec to represent the source code in three downstream tasks as code authorship identification, code clones detection and code comment generation but resulted with a thread of not generalizing to other tasks.

2.LITERATURE REVIEW

1. Introduction

- Briefly introduce the topic of cognitive agent systems and their application in code retrieval.

- Provide an overview of the importance of efficient code retrieval in software development.

2. Cognitive Agent Systems in Software Development

- Define what a cognitive agent system is and how it operates in the context of software development.

- Discuss the role of cognitive agents in automating and enhancing various tasks in the software development lifecycle.

3. Code Component Retrieval in Software Repositories

- Explain the significance of code component retrieval from repositories for developers and software projects.

- Highlight the challenges faced in finding relevant code components efficiently.

4. Techniques and Algorithms for Code Retrieval

- Survey existing techniques and algorithms used in code retrieval systems.- Include keyword-based search, semantic similarity measures, machine learning approaches, etc.

- Discuss the advantages and limitations of each technique.

5. Cognitive Agents in Code Retrieval Systems

- Present research and studies involving the integration of cognitive agents in code retrieval systems.

- Highlight how cognitive agents enhance the accuracy and efficiency of code component retrieval.

6. Case Studies and Applications

- Provide examples of real-world applications or case studies where cognitive agent systems have been applied to retrieve relevant code components.

- Discuss the outcomes and benefits of these implementations.

7. Evaluation Metrics and Performance Measures

- Discuss the metrics used to evaluate the performance of cognitive agent systems in code retrieval.

- Include metrics like precision, recall, F1-score, and user satisfaction.

8. Challenges and Future Directions

- Address the current challenges and limitations in cognitive agent systems for code retrieval.

- Propose potential areas of improvement and future research directions.

9. Comparison with Traditional Code Retrieval Methods

- Compare the performance of cognitive agent systems with traditional code retrieval methods

- Highlight the advantages and disadvantages of both approaches.

3.EXISTING SYSTEM

We have systematically analyzed the reuse problem (especially the cognitive and social difficulties faced by software developers who reuse) from a multidimensional perspective, drawing on our long-term research on information retrieval, human-computer interaction, and knowledge-based systems. Based on this analysis, we propose the concept of *reuse-conducive development environments*, which encourage and enable software developers to reuse through the smooth integration of reuse repository systems and development environments.

4.PROPOSED SYSTEM

We have designed, implemented, and evaluated *CodeBroker*—a reuse-conducive development environment—that autonomously locates and delivers task-relevant and personalized components into the current software development environment. Empirical evaluations of *CodeBroker* have shown that the system is effective in promoting reuse by enabling software developers to reuse components unknown to them, reducing the difficulties in locating components, and augmenting the programming capability of software developers.Almost all types of computer programs logics are readily available and stored at repositories but we don't have any facility to get required component logic in quick simple search, almost for logic search we need to depend on Google which will fetch required component logic with lots of noisy data and its bit difficult for the programmer to get the required logic from that huge noisy data.Detects and counts the number of files according to the type.

5.SYSTEM ARCHITECTURE

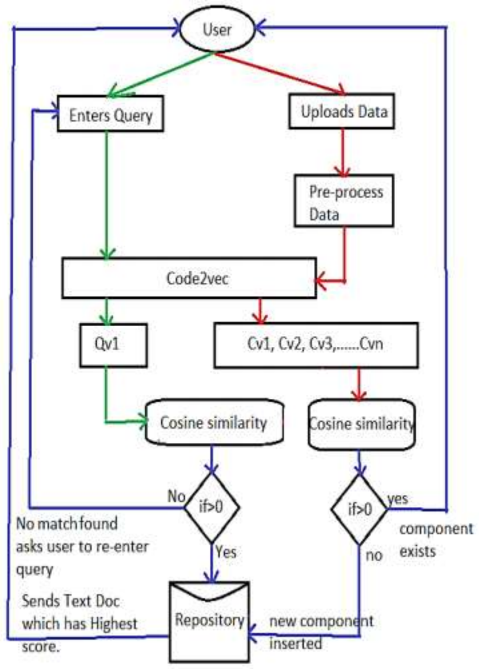
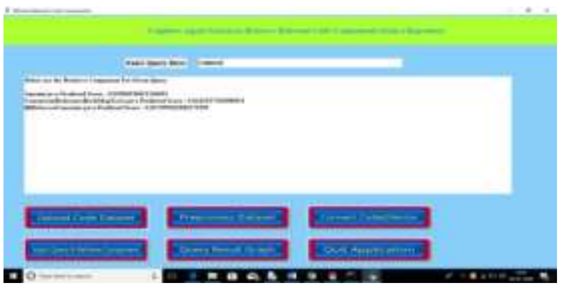


Fig: System Architecture

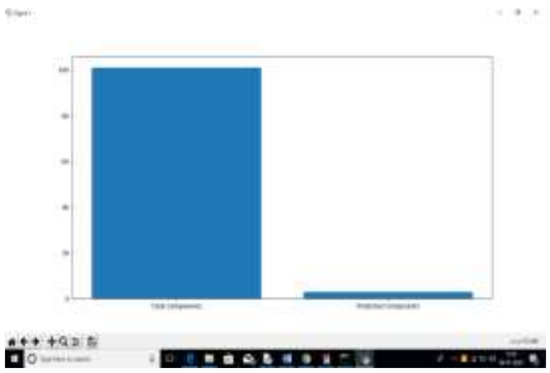
6.RESULT

In above screen I entered query as 'valueof' which means I want to get all component which used or contains logic for valueof function. Below are the search results



In above screen we can see total 3 programs found (Amount.java, ConcurrentReferenceHashMapTests .java and HdfsServerConstants.java) which used or contains logic for 'valueof'. Now I will open 'Amount.java' program from train folder to see whether it really contains that valueof code or not

In above 'Amount.java' from train repository we can see it contains 'valueOf' function. Similarly you can see any query and get component. Now click on 'Query Result Graph' button to get below graph



In above graph x-axis represents total components and predicted components and y-axis represents it count.

7.CONCLUSION

To increase the reuse of software component, software component repositories and to improve the relevancy of the retrieval process - the authors have used Code2Vec concept in which the dataset is embedded by implementing tokenization technique which converts the whole code components from the dataset into vectors. The cognitive system attained success in retrieving the most relevant code snippet by comparing their cosine similarity measure and made it user friendly by abetting in the form of text document.

8.REFERENCES

1. 1. DavidAzcona, Piyush Arora, I-Han Hsiao, Alan Semeaton, "user2code2vec:Embeddings forbProfiling Students Based on Distributional Representations of Source Code", In The 9th
2. International Learning Analytics & Knowledge Conference (LAK19), Mar(2019), Tempe, AZ, USA.ACM,NewYork,NY, SA,10pages.

3. https://doi.org/10.1145/3303772.33038 132. Hong Jin Kang, Tegawende F. Bissyande, David Lo, “Assessing the Generalizability of code2vec
4. Token Embeddings”,34th IEEE/ACM International Conference on Automated Software Engineering (ASE), 11-15 Nov
5. (2019),https://ieeexplore.ieee.org/abstract/document/89524753. Bart Theeten, Frederik Vandeputte, TomVan
6. Cutsem, “Import2vec Learning Embeddings for Software Libraries”,Proceedings of the 16th
7. International Conference on Mining Software Repositories, May (2019) ,
8. https://www.researchgate.net/publication/332300538_Import2vec_-
9. _Learning_Embeddings_for_Software_Libraries4. TimvorderBr¨uck, Mare pouly, “Text Similarity .

AUTHOR

Dr. K. Jaya Rajan professor department of CSE Mallareddy Engineering College for Women,maisammaguda,Hyderabad.

jayarajinfoster@gmail.com