NAME >>>> ARBAZ KHAN

ROLL NO >>>> 2K18/CSME/6

SUBJECT >>>> SOFTWARE ENGINEERING

TEACHER >>>> GULSHER LGHARI

GITHUB LINK >>>>
https://github.com/Arbazkhan123/2K18-CSME-6-ARBAZ-KHAN

TITLE: CHALLENGES IN CHATBOT DEVELOPMENT: A STUDY OF STACK OVERFLOW POSTS

AUTHOURS: Ahmad Abdellatif,, Khaled Badran,, Diego Costa,,

Emad Shihab ,, Rabe Abdalkareem...

PUBLISH: In 17th International Conference on Mining Software Repositories (MSR '20), October 5–6, 2020, Seoul, Republic of Korea.

INTRODUCTION: Chatbots are computer programs designed to hold conversations with users using natural language [15]. Some of them have human identities and personalities to make the conversation more natural. Ranging from Twitter bots with random responses to more complex counseling service agents, chatbots have become increasingly common in recent years. According to Tsvetkova et al. [16], nearly half of the online interactions between 2007 and 2015 involved a chatbot..

RESEARCH: When talking about human-computer conversation, the very first technologies like ELIZA (1966) and ALICE (1995) come to mind

due to their importance. ELIZA was created to demonstrate that natural conversation with a computer was possible, but failed to pass the Turing test since its implementation was based on string matching and no context was taken into account for the written responses. ALICE introduced the famous AIML (Artificial Intelligence Markup Language),which uses pattern matching rules to give 'meaning' to the actual words: topic..

METHDOLOGY: The main goal of our study is to examine what chatbot developers are asking about. To achieve this goal, we resort to analyze the developers' discussions on Stack Overflow as it provides a rich dataset and have been used by similar investigations in other domains, such as concurrency [6], cryptography APIs [41], and deep learning [30].

SUMMARY: More than 50 years after Weinzebaum introduced the first computer program to have a conversation with humans , chatbots have become the main conduit between humans and services . Potentialized by the recent advances in artificial intelligence and natural language processing chatbots are the primary interface in a variety of services, from smart homes and personal assistants ,to health care and E-commerce Given how chatbots reduce the operational costs of services, the usage of chatbots will only increase - experts predict that 85% of users' interactions with services will be done through chatbots by 2021.

MOTIVATION: Businesses constantly need to evolve and adopt newer trends to succeed. These days companies are implementing chatbots that help in solving customer queries, improving communication, and remote troubleshooting to enhance customer experience.

RESULT: **In this paper, we analyze Stack Overflow posts to identify the most pressing issues facing chatbot development. We find that developers discuss 12 chatbot-related topics that fall under five main categories, namely Integration, Development, NLU, User Interaction, and User Input. Chatbot developers are highly interested in posts that are related to chatbot creation and integration into websites. On the other hand, training the NLU model of the chatbot proves to be challenging task for developers. We also find that chatbot practitioners show considerable interest in understanding the behavior of NLUs, while also seeking good recommendation regarding chatbot development platforms and best practices. We believe that our results are useful to the chatbot community as they guide future research to focus on the more pressing and difficult aspects of chatbot development.**

NAME >>>> ARBAZ KHAN

ROLL NO >>>> 2K18/CSME/6

SUBJECT >>>> SOFTWARE ENGINEERING

TEACHER >>>> GULSHER LGHARI

GITHUB LINK >>>>

HTTPS://GITHUB.COM/ARBAZKHAN123/2K18-CSME-6-ARBAZ-KHAN

TITLE: ***SOFTMON: A TOOL TO COMPARE SIMILAR OPEN-SOURCE SOFTWARE FROM A PERFORMANCE PERSPECTIVE***

Authors: *Shubhankar Suman Singh*, *Smruti Ranjan Sarangi*

PUBLISH: **Tue 30 Jun 2020 11:36 - 11:48**
at MSR:Zoom - Quality Chair(s): Jens Krinke

Introduction: ***In this research he work on compare large code-bases and automatically finds the reasons and use the insights to improve the performance of a target applications The classical problem in computer science to detecting two pieces of code are same or not the are three broadclasses to detect the code similarity these are (same, structural, functional) he further classified parposal two broadclasses static and dynamic the static method only use source code and binary stability but the dynamic parameter use to observe***

the behavior such as memory state then he further work on level based categories these are (code version compile optimizers, cross architecture) then add a new categories in this taxonomy different implementation(A softmon tool) where he compare the two codes that represent highly level algorithms yet are coded differently

Methodology: ***he discuss the different class to solve the problem of clone detection the classes are (Graph isomorphism, frequent sequence mining , longest common subsequence static and dynamic features ,comment sequence and contextual information) the softmon tool applies in this category A SOTFMON Tool can comprises 6 components:***

1: A Trace collector: ***to generate the sequence of function cells invoked the execution a program and construct a function call tree from trace***

2: A classification and clustering: ***to classify the call trees into high level tasks and then the cluster the different call trees into fewer groups:***

3: A Graph Engine: ***To compress the and filter the function call trees.***

4: A Annotation: ***To annotated the call trees with there relevant comments.***

5: A Map Engine: ***To find the mappings between the nodes across the call trees of the different applications***

6: A Graph visualization engine: ***To render the mapped trees to simply the human analysis .***

Results: ***In this paper he solve a very ambitious problem to compare the largest open source programs and explan the reasons for their performance difference and he were able to find the diverse set of reason's that explan the most of difrences and able to validate the them against various source and The Softmon tool can take the 200s to complete the analysis the lagrgest code bases .***

Motivation: ***Motivation of softmon tool is compared a diversed set of application, found the 25 reason for the performance difrences, and validate the reason from developers notes and documentation , manual efforts reduce from 1 week to 30 minutes to compare software***

NAME >>>> ARBAZ KHAN

ROLL NO >>>> 2K18/CSME/6

SUBJECT >>>> SOFTWARE ENGINEERING

TEACHER >>>> GULSHER LGHARI

GITHUB LINK >>>>
https://github.com/Arbazkhan123/2K18-CSME-6-ARBAZ-KHAN

TITLE: **Bugine: a bug report recommendation system for Android apps**

AUTHOURS: **Ziqiang Li , Shin Hwei Tan..**

PUBLISH: **n 42nd International Conference on Software Engineering Companion (ICSE '20 Companion), October 5–11, 2020, Seoul, Republic of Korea.**

INTRODUCTION: **Many automated test generation tools were proposed for finding bugs in Android apps. However, a recent study revealed that developers prefer reading automated test generation cased written in natural language. We present Bugine, a new bug recommendation system that automatically selects relevant bug reports from other applications that have similar bugs.**

RESEARCH: **We evaluate Bugine on five open-source Android apps. Table 2 lists information**

about the evaluated apps. We select these apps because they are diverse in app categories, sizes, popularity, and the number of issues.

METHDOLOGY: Building a database of GitHub issues. Our crawler selects Android apps based on: (1) the users' rating and downloads in the App store, (2) the number of discussion and comments by developers, (3) the number of the star and issue of GitHub repository, and (4) the category of GitHub repository. Data Pre-Processing. Extracting app description files. Similarity Measures. Ranking relevant GitHub Issues.

SUMMARY: Bug finding is a creative and inspiring activity. Many automated test generation [8] and repair techniques have been proposed to ensure the reliability of Android apps [1, 3, 6]. However, reading and reproducing the automatically generated test cases could be time-consuming. A study showed that developers prefer reading automatically generated test cases written in natural language [4]. This study also revealed that developers prefer manual testing compared to

automated testing due to the learning curve of automated tools or lack of specific knowledge. Moreover, automated testing techniques for Android apps mostly focus on finding crashes [5], but neglect other non-crash related bugs (e.g., UI bugs). Meanwhile, many manually crafted bug reports (in natural language) are available in open-source repositories like GitHub.

MOTIVATION:We need to make more tools OR APP for people like automate bug fixing,which is help the people to finding a bug in there application and easly solve there bug in application….

RESULT: We introduce a new approach that recommends relevant GitHub issues for an app under test. Given an app under test, Bugine searches for relevant GitHub issues based on the similarities of UI components shared with other apps in our database and further ranks them based on their quality. Our evaluation shows that it helps to discover 34 new bugs in the five evaluated apps.