A review of NLP oriented automated test case generation framework in testing

Ms. Dipti Belsare, Dr. Manasi Bhate
Assistant Professor
Institute of Management & Career Courses (IMCC), Maharashtra, India
Associate Professor
Institute of Management & Career Courses (IMCC), Maharashtra, India

Abstract

Natural language processing is a branch of computer science and artificial intelligence which is concerned with interaction between computers and human languages. Software testing is an activity to ensure a certain degree of quality in software systems. There are many tools available in the test automation industries that are being used to automate test cases and workflows. Natural language programming techniques are continuously getting researcher's attention to automate various software development activities like test cases generation. This paper focuses a light on literature review of how Natural Language Processing is used in generation of automated test case from functional requirement documents or from manual test cases. and how it will improves accuracy and the performance of the automation testing. NLP in software testing can also help to reduce the manual efforts and time to write the test script.

Keywords: *Software testing, Natural language Processing (NLP), automated test case.*

1. INTRODUCTION

Software testing is an activity to ensure quality in software systems. It is an important but expensive activity of software development lifecycle. It is used to strengthen the quality of the product before delivering it to the client. However, software testing is costly. More than fifty-two percentage of the entire development cost is accounted for testing. Hence, it is necessary to control the cost of the testing process because of its time-consuming nature.

Testers conduct most of the phases of software testing manually. One of the phases is test-case design in which the human tester uses written (formal) requirements, written often in natural language (NL), to derive a set of test cases [1]. There are many approaches proposed in the different literatures to reduce these manual efforts for conversion of natural-language requirements into automated test cases using NLP.

Natural Language Processing (NLP) is a field of computer science, artificial intelligence, and linguistics. Instead of writing test cases manually, it investigates a practical solution for automatically generating test cases using natural language [1]. In this paper we referred papers based on most of the NLP techniques, approaches and tools which could use in software testing to generate automated test case from software requirements.

The paper is structured as follows. Literature review is presented in section 2. Findings and discussions of this literature review in section 3. Conclusion in section 4.

2. LITERATURE REVIEW

The goal of the study is to present the review of the literature done in the era of NLP in software testing based on Methodologies, Types of NLP approaches, and NLP tools used to generate automated test cases.

[3] Anurag Dwarakanath and Shubhashis Sengupta propose a tool called Litmus. It does the auto-generation of Functional Test Cases from FRDs written in English. It works on each requirement sentence and generates one or more test cases. There are five steps in the model the first step analyses the sentence through parser and identify whether it is testable. In second step a complex testable sentence is split into simple sentences. In third step test intents are generate from each simple sentence and mapped to the aspects on which the requirement has to be tested. In fourth step generate positive test cases. In fifth step it also it generates negative test cases if applicable. Parser used for parsing is Link Grammar Parser (LG). Best highlight of the paper is first there is no restriction on the structure of the sentence is imposed other thing is that it is tested over various actual requirement documents from various industry projects viz Pharma, IT and Telecom. And the author mentioned the tool gives 77% accuracy over Industrial project requirements.

[4]In this research paper author proposed an approach to support automatic derivation of manually executable test cases from use cases. Author evaluated this approach on industrial case studies like the oil/gas and avionics domains. They worked only on restricted NL and Restricted Use Case Modelling (RUCM) and Restricted Test Case Modelling (RTCM) approach. The transformation from RUCM specifications to TCSs (Test Case Specifications) in RTCM and test cases in RTCM using a set of transformations implemented in the tool called a Toucan4Test. Author classified their work in requirements-based testing, keywords-based testing and behaviour-based testing.

[5]Author proposed the approach to reduce the time and effort of manual testing. The model makes an automatization of BDD (Behaviour Driven Development) test case to UML Class and diagrams, UML Sequence diagrams and code skeleton. Class diagrams generate code skeleton and sequence diagram which can be used to generate step definitions in the desired language. Overall approach improves the efficiency of Behaviour Development approach. The model has implemented this approach through case study for candy machine (vending machine) on the tool called Cucumber in Ruby and used Stanford parser to process sentences. The model works on the restricted NL.

[6] SRQAS- Using two major techniques Natural Language Processing and Information Extraction author has developed a system Software Requirement Quality Assurance System (SRQAS) which is able to assure the quality of software requirements based on testability and non-testability metrics by performing syntactic and semantic analysis on requirements.

3. FINDINGS AND DISCUSSION

After rigorous literature review, we found that many of the research is focusing on the different aspect of software testing.

[3] We found there are some limitations of the tool, first it cannot handle requirements that span multiple sentences and use cases. Second is the less accuracy.

[4] The best part of model is that it gives 100% accuracy as every artefact in restricted way. And the limitations are first both use cases and test cases are specified in restricted NLs. Second Applicability

of this model is only limited to industrial systems from the avionics and oil/gas domains. Third in this paper they have Restricted Test Case Modeling approach (RTCM) focused on “3-transformation”. The editors for RUCM and RTCM have been implemented in a modelling framework called Lightweight Modeling Framework (LMF [4]), similar to Eclipse Modeling Framework (EMF) except that LMF reduces the tight coupling with Eclipse to ease transformations to other platforms. They have not mentioned about NLP approach. The author studies this approach using some case studies like Banking System, Crisis Management System (to manage car crash crises), Autopilot system (controls and guide and air craft), Sub Sea control system (used in managing the exploitation in oil and gas production field). Author summarizes some benefits of the toucan4 model is 1. Systematic and automated 2. Precise and easy to understand 3. easy test case maintenance, 4. Separation of concerns (as use case repository and test case repositories). Limmitation od this study is it related to industry projects.

[5] Overall approach improves the efficiency of Behaviour Development approach. The model implemented on the tool called Cucumber in Ruby and used Stanford parser to process sentences. The model works on the restricted NL. Sanford parses sentences in different languages and returns a phrase structure tree (PST) representing the semantic structure of the sentence (grammar) and typed dependencies (semantic correlation between words in the sentence)

[6] SQARS is developed using JAPE (Java Annotation Patterns Engine) grammar rules to extract thematic roles and detect non-testability features of software requirements. The model involves phases like text pre-processing, text chunking, thematic roles extraction, and Non-testability application.

In all the above studied survey we found the NLP can be used in software testing. There are different NLP parsers available. NLP can improve the performance and efficiency of manual and automation testing by various ways, we can use NLP in automation testing to write automated test cases.

4. CONCLUSION

With this survey paper we found that NLP based automated tests can improve the performance of automation testing. Further improvement can be done in this era by improving the accuracy of the system; we can also evaluate the accuracy and the performance of the system by using different NLP Tools.

5. REFERENCES

1. N. Indurkhya and F. J. Damerau, Handbook of natural language processing. CRC Press, 2010.

2. Muhammad Abid Jamil, Muhammad Arif, Normi Sham, Awang Abubakar, Akhlaq Ahmad, Software Testing Techniques: A Literature Review, Nov 2016.

3. A. Dwarakanath, and S. Sengupta. "Litmus: Generation of test cases from functional requirements in natural language." International Conference on Application of Natural Language to Information Systems, pp. 58-69, 2012.

4. M. Zhang, T. Yue, S. Ali, H. Zhang, and J. Wu, "A systematic approach to automatically derive test cases from use cases specified in restricted natural languages," International Conference on System Analysis and Modeling, pp. 142-157, 2014.

5. Mathias Soeken, Robert Wille, and Rolf Drechsler, “Assisted Behavior Driven Development Using Natural Language Processing” 2013.

6. Bolanle F.Oladejo, Tosin Esther Olorunnishola, “A Natural Language Processing Based Model for Quality Assurance of Software Requirements Document” 2017.

7. Vahid Garousi, Sara Bauer, Michael Felderer, “NLP-assisted software testing: A systematic review” 2019

8. A. Goffi, A. Gorla, M.D. Ernst, and M. Pezzè, "Automatic generation of oracles for exceptional behaviors," International Symposium on Software Testing and Analysis, pp. 213-224, 2016.