VALID 2023

The Fifteenth International Conference on Advances in System Testing and Validation Lifecycle

ISBN: 978-1-68558-101-5

November 13th – 17th, 2023

Valencia, Spain

VALID 2023 Editors

Jos van Rooyen, Huis voor softwarekwaliteit, Nederland

Sandra Viciano Tudela, Universitat Politecnica de Valencia, Spain

VALID 2023

Forward

The Fifteenth International Conference on Advances in System Testing and Validation Lifecycle (VALID 2023), held on November 13 - 17, 2023 in Valencia, Spain, continued a series of events focusing on designing robust components and systems with testability for various features of behavior and interconnection.

Complex distributed systems with heterogeneous interconnections operating at different speeds and based on various nano- and micro-technologies raise serious problems of testing, diagnosing, and debugging. Despite current solutions, virtualization and abstraction for large scale systems provide less visibility for vulnerability discovery and resolution, and make testing tedious, sometimes unsuccessful, if not properly thought from the design phase.

The conference on advances in system testing and validation considered the concepts, methodologies, and solutions dealing with designing robust and available systems. Its target covered aspects related to debugging and defects, vulnerability discovery, diagnosis, and testing.

The conference provided a forum where researchers were able to present recent research results and new research problems and directions related to them. The conference sought contributions presenting novel result and future research in all aspects of robust design methodologies, vulnerability discovery and resolution, diagnosis, debugging, and testing.

We welcomed technical papers presenting research and practical results, position papers addressing the pros and cons of specific proposals, such as those being discussed in the standard forums or in industry consortiums, survey papers addressing the key problems and solutions on any of the above topics, short papers on work in progress, and panel proposals.

We take here the opportunity to warmly thank all the members of the VALID 2023 technical program committee as well as the numerous reviewers. The creation of such a broad and high quality conference program would not have been possible without their involvement. We also kindly thank all the authors that dedicated much of their time and efforts to contribute to VALID 2023. We truly believe that thanks to all these efforts, the final conference program consists of top quality contributions.

This event could also not have been a reality without the support of many individuals, organizations and sponsors. We also gratefully thank the members of the VALID 2023 organizing committee for their help in handling the logistics and for their work that is making this professional meeting a success. We gratefully appreciate to the technical program committee co-chairs that contributed to identify the appropriate groups to submit contributions.

We hope the VALID 2023 was a successful international forum for the exchange of ideas and results between academia and industry and to promote further progress in system testing and validation. We also hope that Valencia provided a pleasant environment during the conference and everyone saved some time for exploring this beautiful city

VALID 2023 Steering Committee

Lorena Parra Boronat, Instituto Madrileño de Investigación y Desarrollo Rural, Agrario y Alimentario and Universitat Politecnica de Valencia, Spain

Yan-Fu Li, Tsinghua University, China

Jos van Rooyen, huis voor software kwaliteit, the Netherlands
Pedro Vicente Mauri, IMIDRA, España

VALID 2023 Publicity Chairs

Lorena Parra Boronat, Universitat Politecnica de Valencia, Spain
Sandra Viciano Tudela, Universitat Politecnica de Valencia, Spain
Jose Miguel Jimenez, Universitat Politecnica de Valencia, Spain

VALID 2023

Committee

VALID 2023 Steering Committee

Lorena Parra Boronat, Instituto Madrileño de Investigación y Desarrollo Rural, Agrario y Alimentario and Universitat Politecnica de Valencia, Spain
Yan-Fu Li, Tsinghua University, China
Jos van Rooyen, huis voor software kwaliteit, the Netherlands
Pedro Vicente Mauri, IMIDRA, España

VALID 2023 Publicity Chairs

Lorena Parra Boronat, Universitat Politecnica de Valencia, Spain
Sandra Viciano Tudela, Universitat Politecnica de Valencia, Spain
Jose Miguel Jimenez, Universitat Politecnica de Valencia, Spain

VALID 2023 Technical Program Committee

Bestoun S. Ahmed, Karlstad University, Sweden
Lilas Alrahis, New York University Abu Dhabi (NYUAD), United Arab Emirates
Francesco Angione, Politecnico di Torino, Italy
Sajid Anwer, Griffith University, Brisbane, Australia
Vincenzo Arceri, University of Parma, Italy
Sadia Azam, University of Verona, Italy
Deepika Badampudi, Blekinge Institute of Technology, Sweden
Sebastien Bardin, CEA LIST, France
Andrea Baruzzo, Interaction Design Solutions / University of Udine, Italy
Davide Basile, ISTI CNR Pisa, Italy
Ateet Bhalla, Independent Consultant, India
Bruno Blaskovic, University of Zagreb, Croatia
Gabriele Boschi, Intel, Italy
Hanifa Boucheneb, École Polytechnique de Montréal, Canada
Laura Brandán Briones, FaMAF | Univ. de Córdoba, Argentina
Mark Burgin, University of California Los Angeles (UCLA), USA
Arjun Chaudhuri, Duke University, USA
Peter Clarke, Florida International University, USA
Bruce Cockburn, University of Alberta, Canada
Aleksa Damljanovic, Politecnico di Torino, Italy
Hichem Debbi, University of M'sila, Algeria
Giorgio Di Natale, TIMA - CNRS / Université Grenoble-Alpes / Grenoble INP UMR 5159, France
Luigi Dilillo, CNRS - IES (Institut d'Electronique et des Systèmes) - University of Montpellier, France
Amelia Dobis, ETH Zürich, Switzerland

Copyright Information

For your reference, this is the text governing the copyright release for material published by IARIA.

The copyright release is a transfer of publication rights, which allows IARIA and its partners to drive the dissemination of the published material. This allows IARIA to give articles increased visibility via distribution, inclusion in libraries, and arrangements for submission to indexes.

I, the undersigned, declare that the article is original, and that I represent the authors of this article in the copyright release matters. If this work has been done as work-for-hire, I have obtained all necessary clearances to execute a copyright release. I hereby irrevocably transfer exclusive copyright for this material to IARIA. I give IARIA permission or reproduce the work in any media format such as, but not limited to, print, digital, or electronic. I give IARIA permission to distribute the materials without restriction to any institutions or individuals. I give IARIA permission to submit the work for inclusion in article repositories as IARIA sees fit.

I, the undersigned, declare that to the best of my knowledge, the article is does not contain libelous or otherwise unlawful contents or invading the right of privacy or infringing on a proprietary right.

Following the copyright release, any circulated version of the article must bear the copyright notice and any header and footer information that IARIA applies to the published article.

IARIA grants royalty-free permission to the authors to disseminate the work, under the above provisions, for any academic, commercial, or industrial use. IARIA grants royalty-free permission to any individuals or institutions to make the article available electronically, online, or in print.

IARIA acknowledges that rights to any algorithm, process, procedure, apparatus, or articles of manufacture remain with the authors and their employers.

I, the undersigned, understand that IARIA will not be liable, in contract, tort (including, without limitation, negligence), pre-contract or other representations (other than fraudulent misrepresentations) or otherwise in connection with the publication of my work.

Exception to the above is made for work-for-hire performed while employed by the government. In that case, copyright to the material remains with the said government. The rightful owners (authors and government entity) grant unlimited and unrestricted permission to IARIA, IARIA's contractors, and IARIA's partners to further distribute the work.

Table of Contents

An Exploration of Maven-Based Java Repositories and Their Testing Practices 1
Canol Simsek and Tugkan Tuglular

In-Service Monitoring and Assessment (ISMA) of Autonomous Driving Vehicles with AI based Algorithms 7
Bangarevva Patil, Thomas Corell, and Rudra Narayan Hota

An Exploration of Maven-Based Java Repositories and Their Testing Practices

Canol Simsek
Department of Computer Engineering
Izmir Institute of Technology
Izmir, Turkiye
email: canolsimsekk@gmail.com

Tugkan Tuglular
Department of Computer Engineering
Izmir Institute of Technology
Izmir, Turkiye
email: tugkantuglular@iyte.edu.tr

***Abstract*—With the increasing significance of testing in software development, particularly in gaming and e-commerce, these industries continue to thrive and evolve, ensuring that the software systems powering them are robust, reliable, and capable of delivering an exceptional user experience. The research aims to compare the testing practices of two GitHub fields and automate repository mining, test code scanning, and gathering source code metric processes. The study aims to uncover common testing usage compared to class and method counts in the gaming and e-commerce software repositories. This exploration provides valuable insights regarding overall test coverage on repositories and how test usage affects code quality. An automated tool is developed for repository mining that clones repositories from desired topics. The main objective of this project is to gather the test usage data and source code sizes by creating and using static source code analysis tools to answer if the test usage in terms of test classes and test methods changes by the sizes of the repositories and does testing have a negative correlation with code smells. Our findings align with our test usage and size metrics expectations.**

Keywords-unit testing; repository mining; e-commerce software; game software.

I. INTRODUCTION

The project is motivated to provide insights into developers' knowledge of testing used in industries, how other industries handle testing compared to theirs, and guide how to improve testing practices and determine if these topics can be automated. With the increasing significance of testing in software development, particularly in gaming and e-commerce, these industries continue to thrive and evolve, ensuring that the software systems powering them are robust, reliable, and capable of delivering an exceptional user experience. Testing plays a pivotal role in achieving these objectives by detecting and addressing potential issues, enhancing system performance, and safeguarding against vulnerabilities. By comparing the testing practices of gaming and e-commerce repositories, valuable insights can be gained into the similarities, differences, and overall effectiveness of testing approaches in these domains. This knowledge will benefit developers by providing them with guidance on improving their testing methodologies and empowering stakeholders to make informed decisions regarding software quality assurance. Ultimately, the findings of this project aim to contribute to the continuous improvement of software quality.

This project aims to mine public repositories, analyze their source codes, and test usages. An automated tool, which clones repositories from desired topics, is developed for repository mining. The main objective of this project is to gather the test usage data and source code sizes by creating and using static source code analysis tools. Collected metric data is processed to create tables and graphs to compare the two GitHub topics: gaming and e-commerce. These visuals provide insights into these questions:

- How does the test usage change with the size of the repositories in terms of test classes and overall classes?
- How does the test usage change with the size of the repositories in terms of the test methods and overall methods?

Our approach for these topics is creating repository miner software to clone public repositories of desired topics from GitHub. After that, create and utilize static source code analysis tools that will scan cloned repositories for their test class count, test method count, class count, and method count. Cloned repositories will be uploaded to SonarQube for analysis and inspection by their code metrics. The repositories successfully uploaded to SonarQube are then scanned by our test code analysis tool, which will search the repository for test classes and methods. Following these steps, each project result from SonarQube and our analysis tool will be combined and used to create tables. From these tables, visuals will be created to gain insights about our topics.

The paper is organized as follows: Section II presents the related work. Section III explains the proposed approach. Section IV presents the result and discussion, and the last section concludes the paper.

II. RELATED WORK

CORVIG [1] created a pioneering study about patch coverage. The paper helps to create a well-developed code review with a highly efficient approach. Its main objective is to spot possible errors and bugs in the systems. With this paper, we learn how to create such infrastructure and a deeper understanding of code analysis. This paper helped us to understand both code inspection infrastructure and to create a tracker tool for our repositories [1].

In [2], Hassan et al. presented a brief history of the Mining Software Repositories (MSR) field and showed guidance about recent methods for pinpointing the bugs,

 ISBN: 978-1-68558-101-5

deployment logs, archived communications, or essential aspects of repositories. Then, they discussed the opportunities for what can be performed to improve this field [2].

Williams et al. [3] studied code coverage with their evolution and provided a tool for bug findings on source code. They describe a method to use the source code change history for refining them and searching for bugs. With the bug database that developers and users of the applications can contribute, they applied mining source code materials and checked over them for bug findings. A static source code checker has done this process. The results are more effective using historical data than other static scanners. They indicated that this project needs to expand, not relying on the user and developer bug reports; projects can provide a new set of bug record data with new types of bugs [3].

Gousios and Spinellis [4] provided a valuable study about how data obtained from GitHub is unsuitable for every research aspect and how this data can be used in large-scale projects. They have used a quarriable offline mirror of GitHub API data for this project to pinpoint the pitfall avoidance strategies. They showcased a GitHub API for streaming metadata from repositories for writing, managing, and optimizing complex queries [4].

Cosentino et al. [5] conducted a meta-analysis of 93 research papers on how they handled data mining from GitHub. The research addressed three dimensions of those 93 papers and addressed poor sampling techniques, lack of longitudinal studies, and replicability issues. Improvements can be made to these topics by developer's data and solution sharing and researchers comparing their results with each other. They can also make guides to how they can clone their projects to make comparisons. They believed these steps could make a general change in confidence in GitHub data [5].

When searching or creating a database in GitHub, researchers had problems with GitHub limitations. Sampling Projects in GitHub for MSR Studies (Dabic et al. [6] provided a GitHub Search dataset with 735,669 repositories to address these issues. With GitHub's millions of repositories, a research paper addressed how much of this data is useful. The systems combine many selection criteria to get the most valuable combinations on GitHub [6].

Kalliamvakou et al. [7] studied quality and available data on GitHub. They analyzed how users handle GitHub features and pointed out the difference between actual data and mined data. They pointed out that maybe the biggest problem for data validity is bias to personal use. Nearly 40% of all pull requests do not appear as merged, even though they were, and half of the users do not have public activity. The paper provides recommendations for developers about how to approach the data on GitHub. As a rule of thumb, the best way to identify a project's activeness is to look at its pulls and commits requests, and the committers are more significant than two [7].

Chaturvedi et al. [8] retrieved all the papers from 2004 to 2013 about Mining Software Repositories published in ICSE. They have analyzed the papers that contained experimental tools or techniques for data mining and repository mining. They have categorized the tools used in MSR on the topics of newly developed, traditional data mining tools, prototype states, and current scripts [8].

By the topic of co-evolution, software needs to evolve, or it will become less valuable over time. Studying the co-evolution of production and test code, Zaidman et al. [9] provided three views that combine information from change history, growth history, and test coverage evolution reports. They applied these three views to two open-source projects and one industrial case to make observations. With these points of view, developers can define different co-evolution scenarios. They also indicated that mining a version control system will provide insight into the testing process [9].

In 2005, Mierle et al. [10] assembled over 200 second-year undergraduate repositories. They have implemented a complete system that parses repositories into an SQL database. The paper examined these repositories' student behaviors, code quality, and code metrics by examining individuals working on the same project separately. However, they point out that the performance indicators cannot predict grade performance. Their results suggest that students' habits and code quality have little effect on their performance [10].

Arcuri and Yao [11] introduced a new view to the co-evolution of software programs and test development. Their approach is to competitive evolution so that both software and testing should directly affect each other. Thus, they co-evolve like prey and predators in nature. The framework is based on co-evolution and search-based software testing [11].

Zaidman et al. [12] studied co-evolution to create awareness among developers and managers alike about the following testing process. In the paper named 'Mining Software Repositories to Study Co-Evolution of Production & Test Code,' they have investigated whether production code and the accompanying tests co-evolve by exploring a project's versioning system, code coverage reports, and size metrics and evaluated their results with the help of log-messages and the developers of the systems [12].

Yalçın [13] developed a co-evolution tracker tool for software with acceptance criteria. The thesis contained 21 real-world projects. Projects are analyzed for every updated and co-evolution process that has been documented. They indicated that when considering Semantic Versioning, Major and Minor version updates have a better ratio for test updates. However, for the result, they found out that even considering major and minor updates, the test update to all update count ratio is not always close to 1.0 [13].

With the evolving complexity of software and test methods, a Literature Review on Software Testing Techniques by Jamil et al. [14] discussed the existing and future testing techniques and how they can be more efficient and enchanted. The paper aims to guide developers to understand and develop their current understanding of software testing techniques for both pre- and post-development cycles [14].

Our work differs from above literature in concentrating on two specific domains and compares the projects with top test usage to explore any patterns.

 ISBN: 978-1-68558-101-5

III. PROPOSED APPROACH

The proposed approach is composed of three steps:

1. Cloning GitHub repositories
2. Scanning GitHub repositories for tests
3. Analysis of GitHub repositories with SonarQube
4. Data analysis

The first three steps are explained in detail in this section. The fourth step is presented in the following section with results and discussion.

A. Cloning GitHub Repositories

Repositories are cloned from GitHub by a repository mining tool. This tool works by sending GET requests with GitHub API. This tool loops through GitHub search pages and projects to decide if the projects provide the requested aspects. If these aspects are provided, it clones them to the local library and creates a folder for them. Here is the example API search line:

'https://api.github.com/search/repositories?q=topic:gaming+language:java&sort=stars&order=desc&per_page=100&page=1'

'/search/repositories': Base URL for the repository search. Provides access to GitHub functionalities. The query component of this URL, denoted by '?q=topic: gaming+language: java': is searching for repositories that are tagged in gaming topic ('topic: gaming') and written in Java ('language: Java'). The parameter '&sort=stars' shows the results are arranged based on the number of stars they've received, which is a good indicator of the popularity among the GitHub. In addition, '&order=desc' makes sure these sorted results are presented in a descending order, meaning repositories with the highest number of stars appear first. The number of results can be changed by changing the 'per_page' parameter.

In this research, our tool will look at the top one hundred repositories of desired pages to efficiently collect repositories. GitHub will limit the request per second if the process is anonymous. This limit will slow down the search process. To bypass GitHub limitations, a user token is needed. The user can get this token from authentication settings in GitHub. The tool then sends a Get request and retrieves a JSON response. It will be cleaned if the repository name has an invalid character that may cause problems. However, not every search result will be cloned. To efficiently scan the repositories, third-party programs will be in use. These programs will work best for Maven-based projects. These projects will have a characteristic file named pom.xml. The tool will search for the pom.xml file in the repositories. This search is performed by checking the 'get_contents' method provided by the GitHub package. The 'pomCheck' method will take two parameters ('parent_directory and 'repo'). 'list' is a list of repositories from search parameters. It will loop through every repository on the API calls. 'Repo' is an instance of the 'repository' class from the 'GitHub' Package. It refers to the repository for the method that will be examined.

If the search string 'havepom = repo.get_contents(path ='pom.xml')' returns true with no exception, this means the repository has the pom.xml file. The reason behind checking the pom.xml file is that our tool analyzes only Java projects that used Maven as its build tool. Pom.xml is a Maven-specific Project Object Model (POM) XML file that contains project layout and configuration information. After the project passes the pom.xml file check, the repository will be cloned, and a folder will be created by its name in the local directory. This process will loop through searched all repositories. With 100 repositories for each page, the program will take some time to clone all projects. So, it will also count the repositories for the user and give an indicator on the console so that the user will have knowledge about what part of the search process they are in at that current time.

After all the process is completed, there will be olders depending on the user selection of the topics. In this research, there are two topics: Gaming and E-commerce.

B. Scanning GitHub Repositories for Tests

Our static source code analysis tool has been used to get repositories' test usage metrics. The tool is designed to analyze Maven projects to gather test usage data.

The 'os' and 'glob' packages form the backbone for this tool as they have been used to navigate the folders and operating systems and efficiently retrieve files based on specified path patterns. To ensure accurate reading of the file content regardless of the encoding, the 'FileUtils' class uses the 'charset' package. As a universal encoding detector, 'charset' identifies the encoding of a file, thus enabling 'FileUtils' to read the file content without errors.

On the other hand, the 'javalang' package is a critical component for two classes: 'TestClassCounter' and 'TestMethodCounter'. This package, targeted at parsing Java 8 source code, helps convert the code into an Abstract Syntax Tree (AST), reflecting the hierarchical structure of the source code. The 'TestClassCounter' class uses 'javalang' to parse code into AST and iterates through it to count class declarations. If an error occurs, it ensures a graceful fallback, returning 0 and an empty list. The 'glob' package gets files from pathnames or specified file path patterns. It is used to find all test files from the provided path. The 'pandas' package is a data processing package that stores the results from this analysis. The tool contains five classes. The 'Main' class contains the logic of execution. The 'FileUtils' class reads the file content and encoding detection. The 'RepoParser' class finds test files in the given repository. 'TestClassCounter' is used for counting the test class counts in a given Java repository. This class parses the given code into an Abstract Syntax Tree using the 'javalang' package and then iterates through it to find class declarations. If it gives an error, it returns 0 and an empty list.' TestMethodCounter' is like the 'TestClassCounter' class, which counts method counts in each repository. It also provides total lines of code in the test methods.

The 'Main' class initializes a list of directories to be scanned. After the initialization, it looks for git repositories using the 'get_repos()' method from the 'RepoParser' class. It then finds all the repository's test files, opens them, and checks their encodings using 'chart.' The 'FileUtils' class

 ISBN: 978-1-68558-101-5

reads the test codes, and 'TestClassCounter' with ''TestMethodCounter' is called. With these findings, the database is created with methods with their method names, classes that methods are located, files that classes contained, folder paths that these files contained, and each repository with their total test lines of codes.

C. Analysis of GitHub Repositories with SonarQube

SonarQube is an open-source platform [15]. It provides continuing code quality management with static code analysis. It can detect bugs, code smells, security issues, and overall code quality. The reason it has been selected for this research is that it can give us valuable metrics about repository sizes so that these can be compared to understand the relationships between them.

SonarQube can be run on different operating systems. The user must select the operating system they are currently using and run the 'sonar.bat' file in the folder. The program will set up its configurations and launch a local server. By default, it will be 'localhost:9000'. The SonarQube server has the following aspects: A web server that serves as the user interface, a search server that utilizes Elasticsearch, a computation engine for code analysis reports, and a database for storing code metrics and instance configurations. It is helpful to note that, in some instances, there might be multiple Java versions on the user's server. In this case, the user needs to define what Java version will be used in the search process. Instead, it is also possible by setting the environmental path for Java to 'SONAR_JAVA_PATH' in the user's local path settings.

Once the analysis is performed, repository metrics can be received from SonarQube. For this, different automation tools have been developed. This tool is also written in Python and uses 'requests' and 'pandas' packages. It communicates with SonarQube and collects code quality metrics. The desired metrics can be selected, or all can be included. After that, it creates an Excel file to store them. With the SonarQube authentications like previous usage, users must have a URL, Username, and Password. Then, the tool will send Get requests with authentication to SonarQube API to retrieve metric data. It then processes the JSON response to extract the metrics by their domain. It then creates a dictionary named 'metric_domains' where each metric key is a new domain. It then iterates over all repositories. For each repository, it creates a 'project_metrics' dictionary. The results are gathered from JSON and stored in that dictionary. After these steps, the tool opens 'ExcelWriter' with the 'XlsxWriter' engine. Then, for each domain that is keyed, it creates a new sheet in the file.

In our research, the following metrics are evaluated:

- Class count: Number of classes written in the project (without test classes).
- Test Class Count: Number of test classes written in the project.
- Method count: Number of methods written in the project (without test methods).
- Test Method Count: Number of test methods written in the project.

IV. Results and Discussion

With the repository mining, source code scanning, and retrieving data from API finished, metrics tables are created. The goal is to compare the two topics' test usage metrics to their sizes and code quality metrics to their test usages. Linear regression, scatter plots, and cluster graphs is used for these tasks. These methods help us to visualize these topics. Intuitive correlations between the metrics are as follows: A positive relationship is expected between class counts and test class counts. Test usage should also be expanded as codebases expand to reduce bugs, errors, code smells, and unwanted program behavior. A positive relationship is also expected with method counts and test method counts. The reason is the same as the class/test class comparison. Test usage is essential for code development and expansion.

Metrics about gaming software repositories can be seen in Table 1. Table 1 lists class count, test class count, method count, and test method count for the five projects with the highest test class counts. Although the ezyfox-server project has the highest number of tests classes among the projects under consideration, it has comparatively less test method than the base project. The base project has the highest number of test methods. The Open Realm of Stars project has the highest test class-test method ratio. Class counts and test class counts would be expected to be similar in optimal cases. However, for the projects in Table 1, there is barely any correlation. Test usage, i.e., test class count and test method count, can be highly different from one repository to another. The same observation can be made for method and test method behaviors.

TABLE I. Total Classes/Methods and Test Classes/Methods for Gaming Software

GitHub Projects	*Class count*	*Test class count*	*Method count*	*Test method count*
ezyfox-server	663	337	2553	943
base	785	314	3742	4256
Open Realm of Stars	302	190	3471	1245
jeveassets	1024	76	8987	743
mmo	184	14	1633	44

The same metrics are collected for the five e-commerce software projects with the highest test class counts and presented in Table 2. Although the io.spot project has the highest number of test classes and of test methods, there are only 11 test classes and 25 test methods. The IOM Project Bootstrap Archetype project has the highest class count-test class count ratio, whereas the productsv project has the highest method count-method class count ratio. The test usage rate seems independent of repository size. There are repositories that have thousands of methods and still have less than 25 test methods.

 ISBN: 978-1-68558-101-5

TABLE II. TOTAL CLASSES/METHODS AND TEST CLASSES/METHODS FOR E-COMMERCE SOFTWARE

GitHub Projects	*Class count*	*Test class count*	*Method count*	*Test method count*
io.spot-next:spot-framework	430	11	2025	25
microservices-event-sourcing/parent	203	9	479	16
IOM Test Framework	337	6	1667	20
productsv	27	6	73	12
[Tool] IOM Project Bootstrap Archetype	16	4	56	6

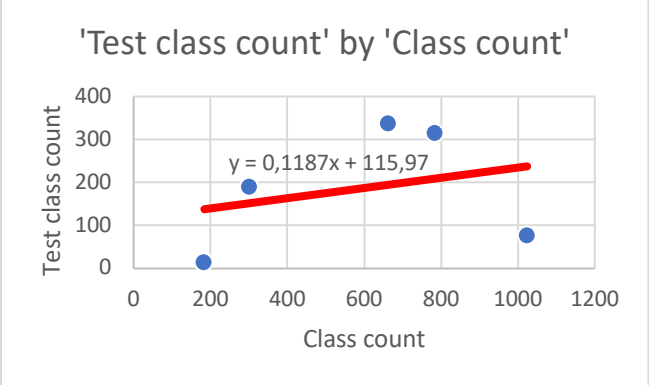

Figure 1. Test Class Count by Total Class Count for Gaming Topic.

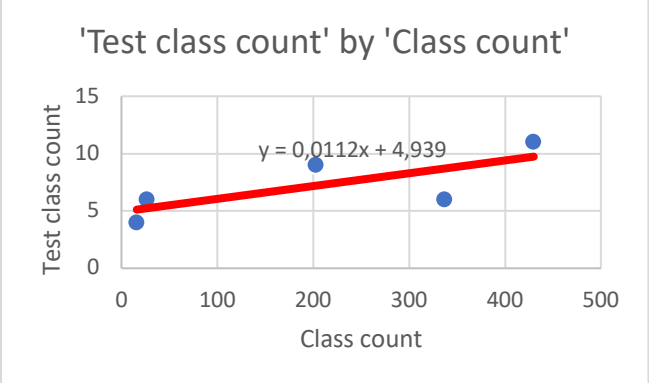

Figure 2. Test Class Count by Total Class Count for E-Cmmerce Topic.

Figure 1 and Figure 2 represent scatter plots of the highest five projects for gaming and e-commerce software projects, respectively. Gaming software projects have higher class and test class counts and higher test class/class ratios. The game software developers tend to write more tests than the e-commerce software developers. Still, both domains need to improve their test development effort.

This project aimed to find answers to these two questions:

- How does the test usage change with the size of the repositories in terms of test classes and overall classes? Answer: Every Java repository has different characteristics. With the Gaming and E-Commerce topics, test usage changes from topic to topic and even between projects. While gaming repositories are much larger than e-commerce repositories, their test usage is unsatisfactory. On the other hand, in e-commerce topic, results are slightly better than gaming. Their repository sizes are pretty small. But they have a better class ratio than gaming topics. So, test usage is expected to have a positive relationship with the class counts in the source code.
- How does the test usage change with the size of the repositories in terms of the test methods and overall methods? Answer: Method counts, and test method usage depend on the software projects' characteristics. It stays the same regardless of the topic. However, in method counts behavior, classes can have multiple functionalities or be small-abstract based. The more methods a class has, the more complicated they can become. It is the same for both source code and testing methods. For our two topics, methods are like class counts in the gaming section, and test usage is the same as expected. In the e-commerce section, classes have fewer methods than gaming topics, and test classes have fewer test methods than gaming topics.

The results of this research cannot be generalized neither to the domains under consideration nor to other domains due to the following reasons:

- the study uses a small sample size, and the results are not statistically significant enough to represent the larger population.
- the study sample might not be representative of the whole population.
- the study uses a non-random sampling method, which can introduce bias.
- the study uses a tool, developed by the authors, that might have some inaccuracies or limitations.

V. CONCLUSION

In this research, software projects from GitHub that are written in Java language have been gathered. Then, they were analyzed by code metrics such as source class/methods and (test class/methods. These processes are automated and can be reused for other Maven-based directories. It turns out that Java repositories may differ quite a lot. Also, test usages are unique from project to project. Test usage is essential for quality products and systems, as more robust tests mean fewer bugs, errors, and unintended behavior. It seems that community projects are being deployed without implementing these principles. Our database extracted dozens of repositories because they had zero test cases.

On the other hand, working with SonarQube and Maven has its own benefits, but these also come with some problems. SonarQube is usually used for developing a project from the start and monitoring it. With already

existing projects, even though it has the requested structure, there might be version and dependency errors with the API and Maven. In conclusion, these processes can be automated using parsers and third-party tools.

For future work, a database of repositories could be expanded by solving these problems mentioned above. For the analysis, more insight might be used for the clustering to gain a deeper understanding of the correlations of the metrics. Moreover, we plan do an in-depth experimentation in order to find correlations of any statistical significance of findings.

REFERENCES

[1] P. Marinescu, P. Hosek, and C. Cadar, "Covrig: a framework for the analysis of code, test, and coverage evolution in real software," The Proceedings of the 2014 International Symposium on Software Testing and Analysis, San Jose CA USA: ACM, Jul. 2014, pp. 93–104. doi: 10.1145/2610384.2610419.

[2] A. E. Hassan, "The road ahead for Mining Software Repositories," The Proceedings of the Frontiers of Software Maintenance, Beijing, China: IEEE, Sep. 2008, pp. 48–57. doi: 10.1109/FOSM.2008.4659248.

[3] C. C. Williams and J. K. Hollingsworth, "Automatic mining of source code repositories to improve bug finding techniques," IEEE Trans. Softw. Eng., vol. 31, no. 6, pp. 466–480, Jun. 2005, doi: 10.1109/TSE.2005.63.

[4] G. Gousios and D. Spinellis, "Mining Software Engineering Data from GitHub," The Proceedings of the IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C), Buenos Aires, Argentina: IEEE, May 2017, pp. 501–502. doi: 10.1109/ICSE-C.2017.164.

[5] V. Cosentino, J. Luis, and J. Cabot, "Findings from GitHub: methods, datasets and limitations," The Proceedings of the 13th International Conference on Mining Software Repositories, Austin Texas: ACM, May 2016, pp. 137–141. doi: 10.1145/2901739.2901776.

[6] O. Dabic, E. Aghajani, and G. Bavota, "Sampling Projects in GitHub for MSR Studies." arXiv, Mar. 08, 2021. Retrieved: July, 2023 [Online]. Available from: http://arxiv.org/abs/2103.04682

[7] E. Kalliamvakou, G. Gousios, K. Blincoe, L. Singer, D.M. German, and D. Damian, "The promises and perils of mining github," The Proceedings of the 11th working conference on mining software repositories, ACM, May. 2014, pp. 92-101.

[8] K. K. Chaturvedi, V. B. Sing, and P. Singh, "Tools in Mining Software Repositories," The 13th Proceedings of the International Conference on Computational Science and Its Applications, Jun. 2013, pp. 89–98. doi: 10.1109/ICCSA.2013.22.

[9] A. Zaidman, B. Van Rompaey, A. van Deursen, and S. Demeyer, "Studying the co-evolution of production and test code in open source and industrial developer test processes through repository mining," Empir. Softw. Eng., vol. 16, no. 3, pp. 325–364, Jun. 2011, doi: 10.1007/s10664-010-9143-7.

[10] K. Mierle, K. Laven, S. Roweis, and G. Wilson, "Mining student CVS repositories for performance indicators," ACM SIGSOFT Softw. Eng. Notes, vol. 30, no. 4, pp. 1–5, May 2005, doi: 10.1145/1082983.1083150.

[11] A. Arcuri and X. Yao, "A novel co-evolutionary approach to automatic software bug fixing," The Proceedings of the 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), Jun. 2008, pp. 162–168. doi: 10.1109/CEC.2008.4630793.

[12] A. Zaidman, B. Van Rompaey, S. Demeyer, and A. van Deursen, "Mining Software Repositories to Study Co-Evolution of Production & Test Code," The Proceedings of the 1st International Conference on Software Testing, Verification, and Validation, Apr. 2008, pp. 220–229. doi: 10.1109/ICST.2008.47.

[13] A. G. Yalçın, "Development of co-evolution tracker tool for software with acceptance criteria," Master Thesis, Izmir Institute of Technology, 2022. Retrieved: July, 2023 [Online]. Available from: https://gcris.iyte.edu.tr/handle/11147/12714.

[14] M. A. Jamil, M. Arif, N. S. A. Abubakar, and A. Ahmad, "Software Testing Techniques: A Literature Review," The Proceedings of the 6th International Conference on Information and Communication Technology for The Muslim World (ICT4M), Nov. 2016, pp. 177–182. doi: 10.1109/ICT4M.2016.045.

[15] "SonarQube Tutorial Documentation," Retrieved: July, 2023 [Online]. Available from: https://docs.sonarsource.com/sonarqube/9.7/extension-guide/web-api/.

 ISBN: 978-1-68558-101-5

In-Service Monitoring and Assessment (ISMA) of Autonomous Driving Vehicles with AI based Algorithms

Bangarevva Patil and Thomas Corell
Autonomous Mobility, Systems & Software Business
Continental Autonomous Mobility Germany GmbH
Frankfurt am Main, Germany
email:{firstname}.{lastname}@continental-corporation.com

Rudra Narayan Hota
AI & Data Engineering, Software Central Technologies
Continental Automotive Technology GmbH
Frankfurt am Main, Germany
e-mail: rudra.hota@continental-corporation.com

Abstract— **Autonomous Driving technology is anticipated to be a key aspect for achieving a higher level of road safety and efficient mobility. A major challenge for this automation is to verify and validate safety aspects of Highly Autonomous Vehicles appropriately. Due to high system complexity and costs, it requires an exponential increase of test efforts for real world testing. Use of In-service Monitoring and Assessment (ISMA) system can efficiently reduce the verification and validation effort with respect to both cost and time. ISMA is an approach to ensure safety of an Autonomous Driving vehicle during the entire product life cycle by continuously intelligently monitoring and evaluating Autonomous Driving functionality during operation. In this paper, we propose an in-service monitoring and assessment concept with a set of exemplary implementations of Artificial Intelligence based algorithms. This concept aims to identify leading indicators for safety critical scenarios and addresses the assumptions made during the development. The research results demonstrate that the proposed approach can safely and efficiently monitor Autonomous Driving functions in both offline and online mode, and helps to discover critical and unknown unsafe scenarios even before an accident occurs.**

Keywords-verification and validation; in-service monitoring and assessment; silent testing; artificial intelligence functions.

I. INTRODUCTION

The development of highly automated driving (HAD) system is making rapid progress, but there is still no satisfying answer on how to prove that autonomous vehicles are safe. The safety validation of HAD system still remains a major challenge. State-of-the-art research on safety validation of highly automated functions (from SAE L3) [1] has found that autonomous vehicles would have to be driven hundreds of millions of miles to demonstrate their reliability in terms of fatalities and injuries compared to human drivers [2]. With such a large number of testmiles , validation of highly automated functions is economically and methodically not feasible [3].

Automated driving (AD) of Society of Automotive Engineers (SAE) L3 levels and above poses much higher requirements for the development and validation of safe systems. We need to improve the way HAD systems are developed and tested. For this, existing design and testing processes need to be extended to processes covering the full product life cycle of a HAD systems. These extended processes enable usage of data collected in the field for continuous improvement of HAD functions [4]. In service monitoring and assessment makes it possible to collect evidence from the field operation to demonstrate that the AD vehicle is safe and remains safe throughout its lifecycle.

In this research work, we introduce a concept called in-service monitoring and assessment, which is a real-time monitoring approach for validating the safe operation of AD systems. In some literature, this approach is also called Silent Testing. Here we describe how Artificial Intelligence (AI) based algorithms can be used to identify leading indicators for a possible accident. For this purpose, we implement prototypes of multiple AI-based event algorithms for pedestrian detection in urban intersection scenarios and evaluate their effectiveness.

The following sections are organized as follows: section II formally describes the problem at hand, section III discussed related work, after-which section IV describes our proposed solutions known as in-service monitoring and assessment, section V describes our example triggers, and finally section VI and VII present our performance evaluation and conclusions.

II. PROBLEM DESCRIPTION

Autonomous driving requires highly complex systems and is expected to be used in an unstructured real-world operational design domains (open contexts) with high levels of uncertainty. In this research, we will focus on how the safety of an automated driving system can be validated against two major challenges.

1. There will be situations or phenomena in the environment that we either cannot predict or are unaware that they may influence the behavior of the vehicle. In other words, there will be “unknown unknowns” events.
   This makes currently considered AI models prone to errors caused by events that are underrepresented in training data. That means, event classes that are safety critical for an AD application are limited, hence it is difficult to accurately evaluate models in such situations.
2. An unstructured real-world operational design domain spawns infinitely many possible scenarios, in which the intended behavior is based on implicit expectations, which are difficult to express formally [5]. For this reason, developing complex systems for open contexts

 ISBN: 978-1-68558-101-5

essentially deals with simplified representations. The validation challenge is closely related to the many necessary simplifications applied during development because every simplification is certainly based on explicit and most often even implicit assumptions. If these assumptions are not justified (even temporarily), the simplified representation is invalid. Therefore, hypotheses need to be formulated and their legitimacy needs to be demonstrated, e.g., through real-world monitoring. In this work, we will be referring to these hypotheses as "trigger functions" of an in-service monitoring and assessment system.

III. RELATED WORK

The impementation of new test strategies for the verification and validation of automated vehicles is required to provide sufficient test coverage while ensuring feasibility and maximum efficiency. State of the Art approaches are expained below.

A. Scenario based testing

In automated systems, research has shown that scenario-based testing can be used to provide an evidence of safety in a manner that is cost-efficient and time-efficient.

In the PEGASUS project [6] a holistic method for scenario-based safety assessment of HAD functions has been developed, using highway chauffeur as an exemplary test object, (i.e., a SAE L3 conditional automation system). The main idea of this approach is to identify relevant scenarios and then generalize them to generate more test scenarios. Scenario based testing exposes the Object Under Test (OUT) to a (pre)defined scenario and the OUT reaction is assessed. In this work, scenarios are derived by combining a data driven and a knowledge-based approach [7]. Scenario-based testing is a promising approach which enables the reduction of the effort required to test a HAD through the identification of relevant scenarios. Nevertheless, the determination of relevant scenarios, the definition of an appropriate parameter space within a scenario and the combination of parameters are great challenges [8].

An AD system will be exposed to a variety of scenarios throughout its deployment lifecycle. As part of the development and assessment process, it is therefore necessary to test against these scenarios. However, it would not be possible to perform such a large number of scenarios in the real world. The use of virtual test environments (i.e., simulations) to perform these tests is therefore essential.

B. VAFOO Approach

A new approach introduced by Wachenfeld and Winner [9] is the Virtual Assessment of Automation in Field Operation (VAAFO), which extracts relevant cases from a huge number of kilometers driven in the random nature of the real world. In the VAAFO approach, instead of testing automated driving in an unsafe environment, functionality is tested virtually in real traffic while the vehicle is driven by a human driver. The decisions of the driver contain additional information about the environment and are used as further input to compare it with the information from the perception sensors. This comparison is used for the event-based trigger.

The method uses differences between the trajectories of the OUT and the test vehicle as a trigger for event-based data recording. The recorded data is evaluated offline after the test drive. Hereby, the environment model of the OUT is corrected retrospectively to create a ground-truth environment model. As the simulation is open loop, the behavior of other traffic participants cannot be influenced by the OUT and therefore only short time frames can be simulated when the behavior of the OUT is different to the test vehicle [10].

C. Shadow mode testing

The shadow mode approach is proposed by Tesla [11]. The idea is very similar to the VAAFO approach. In shadow mode testing, a vehicle is being driven by a human, receiving data from the sensors but not taking control of the car in any way. Rather, it makes decisions about how to drive based on the sensors, and those decisions can be compared to the decisions of a human driver. Both the recorded data and the comparison are used, amongst others, to discover unthought of edge and corner cases, and to evaluate and demonstrate the safety of autonomous functionalities. Based on this technique, shortcomings in the system could be identified, and the collected data can in turn be used to improve their camera-based machine learning significantly. This approach is also used for a fleet of vehicles with human driver behavior as reference system.

IV. IN-SERVICE MONITORING AND ASSESSMENT AS A NEW TEST METHOD

To ensure the safe behavior of automated vehicles, the challenges of the open traffic context must be considered throughout the whole product life cycle. In-Service Monitoring and Assessment will collect evidence from the field operation to demonstrate that the ADS continues to be safe when operated on the road and thus supporting the safety argumentation [12] by addressing the following:

- the dynamic nature of road transportation,
- the assumptions made during development,
- the residual risk of unknown unsafe events.

In-Service Monitoring and Assessment can be used as part of Safety Management System (SMS) [13], DevOps and Learning-driven product life cycle processes for highly automated vehicles [14].

A. In-Service Monitoring and Assessment Framework

An ISMA system provides a framework that allows the performance of a driving function to be evaluated during operation. Figure 1 illustrates the framework using the following steps: a) data acquisition-provides all the data required to evaluate the functionality b) allows the recording & storage of data c) allows the transfer of stored data to a cloud for data management & analysis d) provides an interface that allows an operator to configure the system.

 ISBN: 978-1-68558-101-5

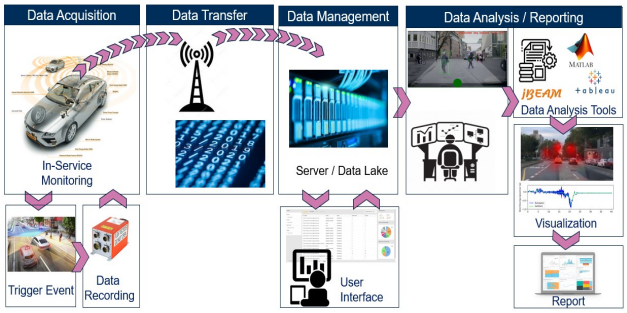

Figure 1. ISMA Dataflow Framwork

B. *Monitoring and Data Collection Approaches*

One possibility to reduce the amount of data and the analysis effort is real-time data monitoring during operation. Basically, the two different methods that can be used to trigger data collection are rule-based approach (e.g., a threshold exceeding case analysis) and data driven approach [15]. Data collection depends upon multiple different factors and their possible alternative choices as shown in the below Table 1. For many data collection pipelines presented in literature, the dependent factors are theirs inputs, along with the types of said inputs, the applications being considered, trigger function definitions, and reference systems. And for each of these factors, there are different alternatives already applied in various applications. Some of them are mentioned in the different rows of the Table 1. As monitoring and collecting data mainly depends upon the reference system, this is the most important factor of all.

TABLE I. DEPENDEMT FACTORS AND ALTERNATE CHOICES FOR MONIOTORING AND DATA COLLECTION.

Inputs	Application	Trigger Function	Reference System
Real world or Simulation Scenarios	Cut-in, Lange Change	Object state & Position	Reference Map
Urban or Highway Scenarios	Object state uncertainty	Driving behavior error	Virtual Scenario
Offline or Online Streams	Driving path and Trajectories	Trajectory deviation	Drivers Input
Single or Fleet of Vehicles	Anomalies & Corner Cases	Corner Cases	Complementary Sensors

1) *Rule based approach:* One possible method for triggering data collection is when the measurements or derived attributes exceeds certain threshold values. Thresholding events are equivalent to leading indicators as they are known in the field of safety assessment.

The advantage with rule-based approach is that it is simple to define and develop. In many cases, they are based on semantics of the scene. Safety critical cases can be triggered with simple rules on derived feature for e.g., setting threshold on the speed of pedestrian crossing the road. It is also possible to use model parameters and model confidence to set the triggers. The challenges with rule-based approaches are in both the selection of relevant subset of rules and in finding appropriate triggers for AD functions. Below are some of the rule-based examples.

a) Lane mark Recognition [16]: In this rule-based approach the estimated lane marks, gathered from sensorics inputs, are compared to equivalent marks from online map data and GPS localization, as shown in Figure 2. Both left and right lane marks are used for this comparison.The trigger is then defined using the deviation between the lane marking recognition and the lane markings retrieved from map data. Selected scenes where the recognition algorithm does not perform well are saved for analysis.

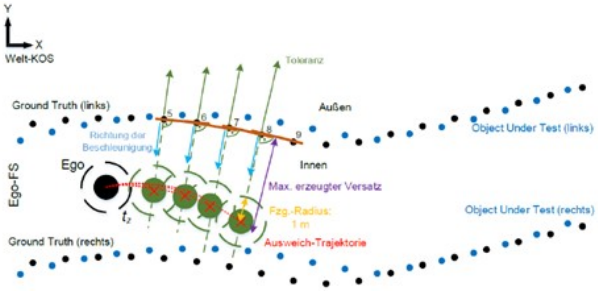

Figure 2. Ground truth (black dots) and estimated lane marks (blue dots) with ego vehi cle position (black and green dots)

b) Trajectory prediction: We used prediction error with pedestrian trajectories. Some of the example cases are shown in Figure 3. A high prediction error with respect to future positions causes a trigger to be generated and can be used to filter out special and atypical cases.

Figure 3. Cases showing mismatch between pedestrian actual future position (green track) and predictions (ged track).

2) *Data driven approach:* This method can be useful for detecting anomalies in a specific scenario by first establishing the profile of a "nominal" vehicle behavior and then mathematically identifying outliers. This method does not require a problem to be known in advance in order to detect the outliers (unlike an analysis of exceedances), but there are also certain limitations. The problems detected are not clearly contextualized and once detected, extensive expert analysis is required to understand patterns of causality. Data driven methods, on the other hand, are easier to implement but lack contextualization. Below is an example of the data driven approach.

a) Success and failure case estimation [17]: This is a data driven approach designed to predict failures as early as

 ISBN: 978-1-68558-101-5

possible by using sensor data from up to ten seconds before each disengagement. In this work different sensor measurements are taken as input for categorization of success or failure cases. They categorize the event into success and failures cases which is very similar to classifying it as normal and anomaly.

V. SUITABLE ISMA TRIGGERS DEVELOPED WITH AI FUNKTIONS

In our examples, we mainly focus on developing our triggers using mostly rule-based approaches along with a few data driven approaches. We will now briefly describe the development of our various example triggers.

A. *Pedestrian detection:*

The pipeline for pedestrian detection and trigger definition is shown in Figure 4. Here the trigger is defined to check presence of pedestrian within the defined Region Of Interest (ROI).

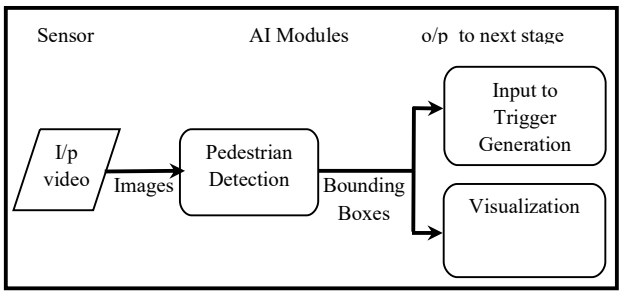

Figure 4. Pipeline for the pedestrian detection on video stream and trigger generation

Some of the active trigger examples with pedestrian detection are shown below in Figure 5, where the pedestrians are withing the defined region of interest, hence satisfying defined triggered condition.

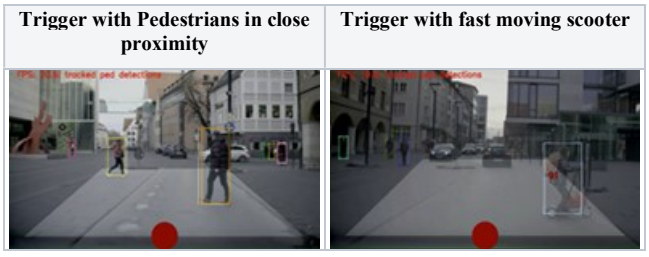

Figure 5. Trigger examples with pedestrian detections

B. *Pedestrian trajectory prediction:*

The algorithmic pipeline for trajectory prediction, which takes both video stream and vehicle speed as inputs to predict future paths, is shown in Figure 6. The inputs to the trajectory prediction are object detections and multiple object trackings. The past positions of tracked objects are used in the algorithm to predicts future paths.

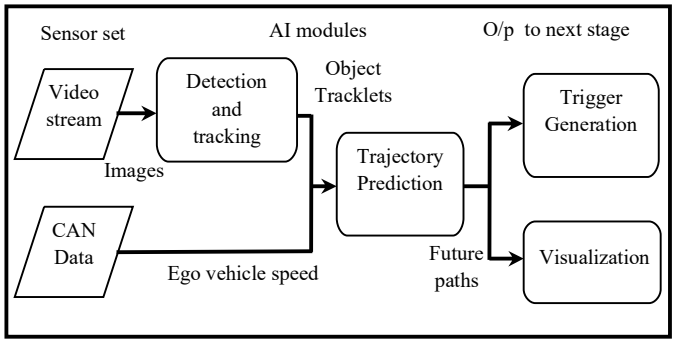

Figure 6. Pipeline for the pedestrian trajectory prediction on video stream and trigger generation.

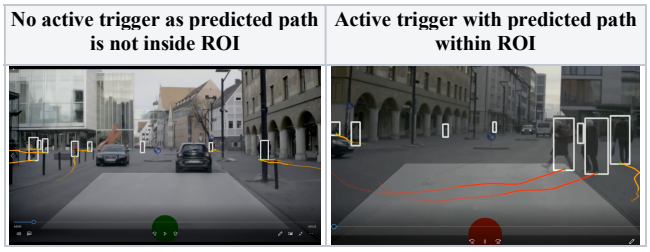

Figure 7. Trigger examples with trajectory prediction

The trigger is then activated using the definition as the predicted position within, and crossing, the center of the defined ROI. Some of the examples are shown in Figure 7.

C. *Event Detection:*

This AI function works on a sequence of input images and estimates the category of events going to happen (pipeline as shown in Figure 8) [18]. The trigger is defined to flag all of these anomaly cases that are considered as rare and unique driving situations, which can further be used for model improvement through continuous updates.

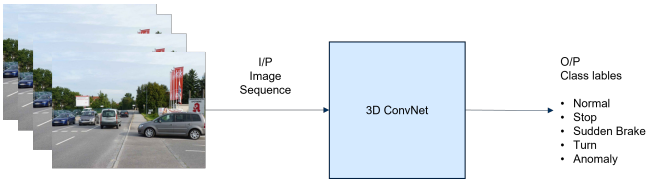

Figure 8. Pipeline for the Event detection on video stream.

Some of the examples, belonging to known event classes, are normal driving, sudden breaking and turning. It also can also predict anomaly events such as: camera falling, heavy jittering due to driving on uneven roads and scenes with low visibility. Examples of event classification and anomaly events are shown in Figure 9.

 ISBN: 978-1-68558-101-5

Figure 9. Examples of Anomaly event classification.

D. *Event Discovery:*

Here we applied an approach called Generalized Event Discovery (GED), which is an extension of "Generalized Object Discovery (GOD)" [19]. This is a fairly new approach to estimate the number of classes in the unlabeled data. This approach can categorize repeatedly occurring new objects or events, by attribute clustering from the unlabeled set. An illustration of this pipeline can be found in Figure 10.

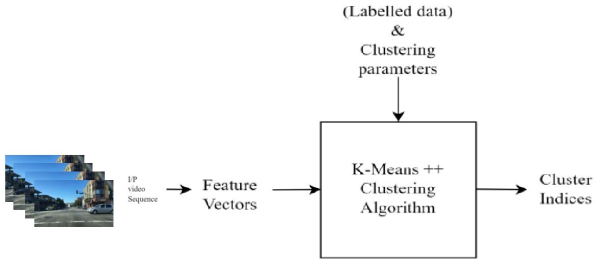

Figure 10. Pipeline for the Event discovery on video stream.

In case of anomaly detection or unknow class of event detection, approaches are focused on the boundary elements with respect to the distribution of the learned data. Furthermore, in GED approach, similar unique events are grouped together, and can later be labelled and used for continuous updates and learning. Figure 11 shows two example situations discovered in the driving data: on the left, a near accident case, and on the right a sudden breaking case is depicted. These kinds of samples can be used as either triggers or filters.

Figure 11. Example of safety critical event cases.

VI. PERFORMANCE EVALUATION AND ANALYSIS

We evaluate our approaches using prepared sub-sequences from BDD100k video data. The total number of sub-sequences are around 14000. The overall classification performance of different event categorization is 79.85%.

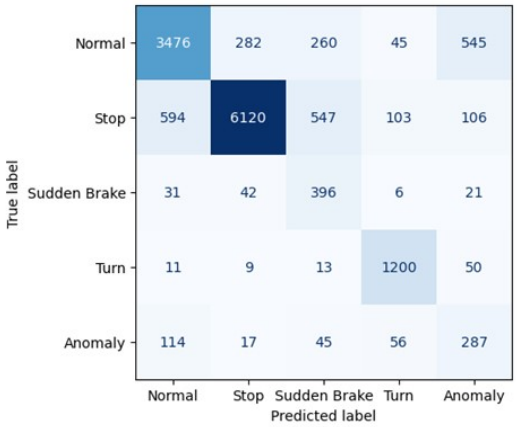

Figure 12. Confusion Matrix for Event detection Performance.

The above confusion matrix plot shows the overall correct classification of event categories. The off-diagonal elements in the table show the different classifications from the true class labels. The top right corner element shows that up to 10% data is categorized as anomaly in comparison to the normal labeled data. These samples can be selected using a defined trigger, some of them can be safety critical in nature. With further analysis we can select out safety critical cases from these and use them for other systems such as self-adaptive systems or for DevOps process for continuous learning.

VII. CONCLUSION

Safety assessment and validation for HAD systems is still very challenging considering the system complexity and cost of deployment. In this paper, we presented an In-Service Monitoring and Assessment approach as a new method for safety validation of automated driving functions under real world conditions. This article covers relevant literature on verification and validation, different approaches for monitoring of HAD Systems during operation and state of the art development of triggers. Along with rule-based approaches, we also covered various data driven monitoring approaches to identify rare, unusual, and critical driving situations.

Future work on this topic includes the exploration of appropriate sets of triggers, defining suitable metrices for their evaluation, and selection of context specific trigger subests. Use of safety critical data for continuous learning and improvement is also a topic that needs further study.

 ISBN: 978-1-68558-101-5

ACKNOWLEDGMENT

The research leading to these results is funded by the German Federal Ministry for Economic Affairs and Climate Action within the project "VVM - Verification and Validation Methods for Automated Vehicles Level 4 and 5". The authors would like to thank the consortium for the successful cooperation.

REFERENCES

[1] SAE International, "Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles," 2018.

[2] N Kalra,and S. M. Paddock,. "Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability?". Transportation Research Part A: Policy and Practice, 94, pp.182-193, 2016.

[3] W. Wachenfeld, and H, Winner, "The new role of road testing for the safety validation of automated vehicles. Automated driving: Safer and more efficient future driving", pp.419-435, 2017.

[4] N. Marko, E. Möhlmann, D. Ničković, J. Niehaus, P. Priller,and M. Rooker, "Challenges of engineering safe and secure highly automated vehicles", arXiv preprint arXiv:2103.03544, 2021.

[5] J. E: Stellet,T. Brade, A. Poddey, S. Jesenski, and W. Branz, "Formalisation and algorithmic approach to the automated driving validation problem", IEEE Intelligent Vehicles Symposium (IV) (pp. 45-51). IEEE, June 2019.

[6] German Aerospace Center: PEGASUS-Project https://www.pegasusprojekt.de/en/home, 2019, Access 22.03.2021.

[7] S. Riedmaier, T. Ponn, T., D. Ludwig, B. Schick, B. and F. Diermeyer, "Survey on scenario-based safety assessment of automated vehicles". IEEE access, 8, pp.87456-87477, 2020..

[8] P. Junietz, W. Wachenfeld, W., K. Klonecki,and H. Winner,"Evaluation of different approaches to address safety validation of automated driving", 21st International Conference on Intelligent Transportation Systems (ITSC) (pp. 491-496). IEEE, November, 2018.

[9] W. Wachenfeld, and H. Winner, H., "Virtual assessment of automation in field operation a new runtime validation method", In Workshop Fahrerassistenzsysteme (Vol. 161), September, 2015.

[10] C. Wang. "Silent Testing for Safety Validation of Automated Driving in Field Operation". In Technische Universität Darmstadt. Dissertation - (Retrieval date: 01.06.2022), 2021.

[11] Tesla: What is Shadow Mode Tesla Autonomy, https://www.youtube.com/watch?v=SAceTxSelTI, 2019.

[12] VMAD-SG3-20-02 Concept and agenda for the VMAD SG3 technical workshop on ADS in-service monitoring and reporting. 17th March 2022.

[13] FAA Order 8000.369C - Safety Management System 2020-06-24, PDF.

[14] M. Haiber, et al. „Learning Driven Product Lifecycle for Automated Driving Systems ", Systems Engineering day. 12.11.2021. https://www.tdse.org/, PDF.

[15] A. Scarinci, "Monitoring safety during airline operations: A systems approach", Doctoral dissertation, Massachusetts Institute of Technology, 2017.

[16] Wadim Tribelhorn "Conceptual design and prototypical implementation of a "Silent Testing" method for automated lane marking detection evaluation for automated vehicles" Master's thesis,TU Damstadt, Nr. 703/18,p. 68, 2018.

[17] C. B. Kuhn, M. Hofbauer, M., G. Petrovic,and E. Steinbach, "Introspective black box failure prediction for autonomous driving". IEEE Intelligent Vehicles Symposium (IV) (pp. 1907-1913). October, 2020.

[18] G. Yu, S. Wang, Z. Cai, X. Liu, C. Xu, C. and C. Wu, "Deep anomaly discovery from unlabeled videos via normality advantage and self-paced refinement", In Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition (pp. 13987-13998), 2022.

[19] S. Vaze, K. Han, A. Vedaldi, and A. Zisserman, "Generalized category discovery". In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 7492-7501), 2022.

[20] C. Wang,K. Storms, and H. Winner, H., "Online safety assessment of automated vehicles using silent testing", IEEE Transactions on Intelligent Transportation Systems, 23(8), pp.13069-13083, 2021.

[21] A. Koenig, K. Witzlsperger, F. Leutwiler, and S. Hohmann, "Overview of HAD validation and passive HAD as a concept for validating highly automated cars",Automatisierungstechnik, 66(2), pp.132-145. 2018.

[22] J. A. Bolte, A. Bar, D. Lipinski, and T. Fingscheidt, "Towards corner case detection for autonomous driving," In 2019 IEEE Intelligent vehicles symposium (IV), pp. 438-445, June, 2019.

 ISBN: 978-1-68558-101-5