IJISAE

International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org Original Research Paper

Controlling Runtime-Anomaly of A Web Application Using Advanced Machine Learning

Partha Pratim Biswas[1*], Amit Dixit[2], Tanupriya Choudhury[3]

Submitted: 15/01/2024 **Revised:** 25/02/2024 **Accepted:** 05/03/2024

Abstract: INTRODUCTION: Controlling run-time anomalies using machine learning in software will help us to increase product efficiency and reliability. In this field, we are still in a very early stage. In our case, for a Web-Application the anomaly detection and correction at run time is yet to be developed.

OBJECTIVES: The goal of this study is to provide a method to Detect and Classify an anomaly and Fix that on run time using Machine Learning.

METHODS: A systematic step-by-step approach is launched with the study of 57 papers on anomaly detection algorithms and Runtime error detection and correction methodologies

RESULTS: The major number of papers are related to network anomaly detection algorithms (42%) and another real-time system (like flight/train, etc.) error detection and corrections techniques (18%). Papers related to software error detection are 30%, and papers our research focused on are 10%.

CONCLUSION: To detect an anomaly in the application log, we would be using the Pattern search and Local Outlier Factor to detect and classify the errors. Then finally, using the Generative AI, we will fix the detected errors.

Keywords: *Runtime anomaly, Machine learning, Web Application Monitoring, Autoencoders, Security*

1. Introduction

The product industry gives support only when there is an issue reported. Always relying on the customer's feedback or report. Nowadays, a very small number of software industries monitor and track the errors in run We can never assure a 100% error-free environment. Still, by implementing such technology into our current infrastructure, we could significantly reduce product failure rates while also providing reliable results with greater accuracy than before - ultimately leading to happier customers who trust their products more than ever. time/production. The encountered issue will only be reported/fixed if the user understands that there is a bug! This may take another few weeks/months

Some of the research has already been done in this scope.

QVM (Quality Virtual Machine) and watchdogs like Comprehensive and efficient runtime checking in system software are a few of them. But they are related to saving that environment (in that specific time) properties so that the scenario can be recreated to fix. To address this issue, researchers are exploring ways to develop plug-and-play error/failure detection systems that utilize machine learning (ML) algorithms for learning from past errors/failures/bugs encountered by similar systems. Once detected using ML techniques, AI-based solutions will try fixing them automatically or prompt notification to the manufacturer so they can provide an appropriate hotfix release.

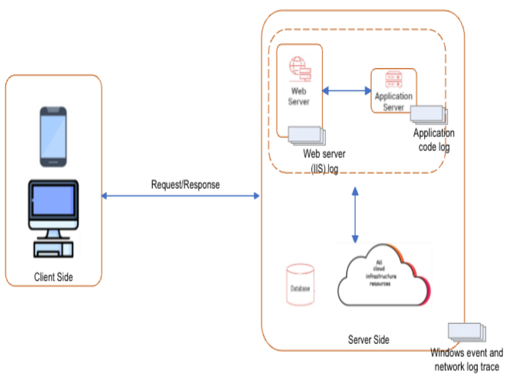

Fig 1. The error will be traced and classified using the logs.

[1*] *Research Scholar, Quantum University, Roorkee, Uttarakhand, 247167, India. ppbiswas82@gmail.com*
[2] *Professor, ECE Dept., Quantum University, Roorkee, Uttarakhand, 247167, India. amitdixit.ece@quantumeducation.in*
[3] *Research Professor, CSE Dept., Graphic Era Deemed to be University Dehradun, 248002, Uttarakhand, India. tanupriyachoudhury.cse@geu.ac.in*

An interactive computer program that has two primary layers - Client and servers. Client-built with web technologies such as HTML, C SS, and JavaScript. The Server is built with Java, C#, Python, and other serverside programming languages. Based on the requirements, the server layer may interact with the Database, Cloud, Operating System resources, or other service APIs, etc.

In this Era, Automation is the only way to do business. Web Application is widely used in such scenarios.

	1. Free University of BozenBolzano, BozenBolzano Italy 2.Università degli Studi dell'Insubria, Varese, Italy	sends that to the central continuous exception monitoring server to raise a ticket to track the issue. Once the developer fixed the issue and all the regression testing is done the SZZ algorithm is to be run to identify the root cause of the exception. Finally the continuous inspection tool checks if any code smell induce the exception report the same as recommenda tion. Recommend ation system builds a quality model using the code smells and their metrices.	medium sized enterprises and a set of prediction techniques such as regression and machine learning. They provided a tailored SQA model to the identified companies to gather the bug/issue's history.	developer organizat ion using an issue tracking tool, say Jira. Here, they did not try to learn the error occurred in end user's site, using the machine learning and no analysis for further action for automatic fixing.
Ruchik a Malhot ra Nov 2014 [2]	A systematic review of machine learning techniques for software fault prediction Delhi Technologic al University	To do timely correction of faults we need an mechanism to detect them early. Using Software Fault Prediction (SFT) the classification of the models as fault prone are not. So that the professionals can identify the testing resources (fault prone) in the early phase of SDLC. By using this concept the tester can perform a	Using ML techniques for SFP models. Secondly, to construct the SFP model consider the performanc e accuracy along with the capabilities of the ML model. Identify the difference between the statistical and the ML techniques. Compare the performanc es on accuracy among the ML techniques.	Here, the research had been done for the early predictio n and detection of the errors/bu gs while in developm ent stage to improve the quality using the ML concept named - SFP (Software Fault Predictio n).
Author / Date	**Topic / Focus / Question**	**Concept Theoretical Model**	**Paradigm Method**	**Research gaps / Findings**
Valenti na Lenard uzzi Et al. [1] Sept. 2017	A Dynamical Quality Model to Continuousl y Monitor Software Maintenanc e	When an stake holder encounter an error, the error would be listen by the "exception listener" on the JVM and	They identified set of software quality assurance those are being practiced in small and	The research was done for to track the exception s and notify the exception to the

2. Literature Review

Software failure at run time, studied in different research. Some of them are Comprehensive and efficient runtime checking in the system, QVM - QVM-quality virtual Machine, and A Dynamical Quality Model to

Continuously Monitor Software Maintenance. In almost all the research the researcher focused on saving the failure configuration so that the scenario can be recreated by the developer and fix them. Some research related to the study related to failure due to resource leaks. And so on.

Different researchers tried to provide models or approaches to predict errors or failures when the software is under development, in other words, early detection of errors. And found some of the research works related to run time software failure detection where the failure may be a memory leak or a resource failure so that the system condition of that specific time can be captured and can be used to recreate the scenario by the developer to fix the issue.

The wide area of our research would be related to Machine Learning and Artificial Intelligence, which covers the following areas:

- Detect the error/failure that occurred in the system that failed the software using soft computing by using the ML algorithm.
- Classify the error type using the Classification algorithm.
- Analysis and learning of different types of run time failures or errors using ML algorithm.
- Check and apply the automated fix (available) based on the analysis using soft computing. If no fix is available, notify us to prepare a fix.

Table 1. The tabular form of Literature Review

		quality test even with the less resources.	By summarizing the strength and weakness of ML techniques identify the application of the ML techniques in SFP.	

				ent and analyse to fix them using AI or initiate fix to the company.
Syahana Nur Et al. [5]	Machine Learning Techniques for Software	The research is used to identify the similar	Systematic Literature Review (SLR) was	This research is about

Awni Hammouri Et al. [3] 2018	Software Bug Prediction using Machine Learning Approach	Software Bug Prediction SBP is based on three supervised machine learning algorithms. Prediction of future software faults is done based on the historical data. To research talks about the increasing the software quality, reliability and reducing the maintenance cost.	Prediction of future software faults is done by using three ML algorithms. Naïve Bayes (NB), Decision Tree (DT) and Artificial Neural Networks (ANNs). The research used the historical fault data, required metrices and computing techniques. By comparing these algorithm	The research was done to predict the software bugs to fix them before production release. The research talks about the future work about to compare more ML algorithm to improve the efficiency of the model. Whereas, our proposal is to track and learn the bug using ML, that are occurring at live/production software environment (customer site). It means our research will provide a model to learn the errors using ML, that occurs at client end at runtime / production environm

Sep 2020	Bug Prediction: A Systematic Review Universiti Putra Malaysia	models those are available to predict the bugs of a software before the testing starts. The researchers narrowed down to 31 main studies on this research. This research was to check, by using machine learning till what extent the bug prediction is possible of a software.	used to conduct this research. Multiple of (31) Model study was considered for this. Six machine learning algorithm had been identified. Area Under Curve (AUC) or F-Measure are bused to evaluate the performance of the models.	identifying the different methods and their accuracy with the methods used to predict the software bugs. By understanding the different methodology and models discussed here we can use one or more ML algorithm to detect the runtime errors/bugs of a live system and learn then to do further analysis and solution.
Matthew Arnold Et al. [4] Oct 2008	QVM: an efficient runtime for detecting defects in deployed systems	Software defects occurs related to configuration (may be some specific configuration that fails the system) in post deployment environment and in such scenario it is impossible to hold the production system to track or recreate the		The research is to track the run time software failure and trace the configuration properties, for what the failure occurred like resource leak. This research does not cover the

		issue to fix them. In such case recreation of the issue in testing environment is a challenging, affordable and time consuming task. To overcome this the research proposed an app roch named "Quality Virtual Machine", which detects the defects by continuously monitoring the production application and validate the user specified correctness of properties such as, Java assertion, heap properties etc. By implementin g the QVM on IBM J9 JVM the researcher could detect and fix multiple issues in several live applications.		runtime exception for other failures like run time code exception s, hardware failure, network failure and the configura tion failure. Our proposed research model is to learn the failures/e rror at the run time in productio n environm ent using ML then analyse and if possible fix that.
Chang Lou	Comprehen	To detect any anomalies	Exploration type of	The researche r

		task. So the research is regarding a software which is a intrinsic failure detectors that detect anomalies with finer granularity. Then precise fault information is saved to reproduce the production failure to help for expedite recovery. They named this as intrinsic watchdogs.	principles for coding - what makes a good watchdog. 3. An effective type of watchdog is one that mimics the mail program - How to write the watchdog checkers. Checker constructio n has three general approaches: 1. Probebased that works like a special client and invokes the software's public APIs with presupplied input. 2. Define some system health indicators and then write a checker to monitor each one. 3. Checker selects important operations from the main program, mimics them and detects errors.	issues occur at run time. Whereas we are learning the runtime failure using the ML algorithm and take action after analysis using AI.

Et al. [6] May 2019	sive and efficient runtime checking in system software through watchdogs	occurs in a system can be tracked from outside by using a thin module with a simple independent underlaid code is enough to monitor the failure. But tracking the intrinsic failures, occur in system software is not a easy	methodolog y has been used initially for this intrinsic failure detector. Some points were considered here are: 1. Define the characteristi c of good watchdog - watchdog Abstraction. 2. Define the design	filled the gap they find by impleme nting the mimic type watchdog automatic ally by using the main program source code. The research is only for the intrinsic

Dr. K. P. Parades hi Et al. [36] Sep 2022	Implementa tion of Fault Detection Framework For Healthcare Monitoring System Using IoT, Sensors In Wireless Environmen t	This research talks about the fault detection monitoring framework for IoT sensors and wireless environment in healthcare. This framework of fault detection identifies the flaws in the system and isolates the complex	According to research, fault can occur in three ways. Device failure, Software issue and Communica tion failure. Failures can be detected: Detect the damaged sensor nodes and select the alternate set of nodes to	The research was about the utilizatio n of IoT and wireless networks for patient health to analyse and detect the fault in monitorin g devices. This

		process or variables so that we can save extra relevant information about the problem. This fault detection framework provides an effective impact in an IoT sensors and wireless environment in healthcare.	continue the data transmissio n. The faulty nodes will be barred from participatin g until they are repaired or replaced. Sensor network should be tiny and human influence should be minimum on installation install in any sort of environmen t. To validate the findings the MSVM model was implemente d and to classify the classes in the dataset the SVM model was used.	research shows 80% accuracy in fault detection. If we learn the hardware , software, configura tion and network failures using the machine learning and try to fix them using the AI may create an impact in quality of the IoT devices and wireless networks. In our research we are trying to fill this gap by working with an

			or faulty. Once the fault is detected, the fault diagnosis is performed at a central node, they are called a network server, to reduce the computatio nal load on the sensor.	
L. Seabra Lopes Et. Al. [8] Aug 1995	A machine learning approach to error detection and recovery in assembly	The research is focusing on error detection and recovery in robotized assembly system in a manufacturin g system.	The proposal are divided into in 3 layers. 1. Planning Strategy 2. The Assembly Domain Knowledge 3. Supervision Functions 4. Training and Learning 5. Experiment al Setup 6. Failure diagnosis	This research is about a manufact uring assembly system error detection and recovery using the ML. But our research is about Software error detection and correctio n using AIML.

				web/app software.
Sana Ullah J an a Et. al. [7] Aug 2019	A distributed sensor-fault detection and diagnosis framework using machine learning	The research is about a distributed sensor-fault detection to diagnose a system based on machine learning algorithms.	Here the fault detection block is implemente d in the sensor in order to achieve output immediatel y after data collection. This block is made up with an autoencoder to transform the input signal into a lowerdimensional feature vector, which is then provided to a Support Vector Machine (SVM) for classificatio n as normal	This research talks about the run time sensors fault detection and diagnosis in IoTs. In our research we are proposin g to detect and learn the software error using ML and fix and notify using AIML.
			that the anomaly query classificatio n can be done to specific attack type.	run time failure os a software and try to fix using AIML.
Laksh mi Geetha njali Manda gondi Et. Al [31] Feb 2021	Anomaly Detection in Log Files Using Machine Learning Techniques	The objective is to use machine learning techniques to "learn" how to distinguish between a log file which has an anomaly.	Here the researcher used multiple algorithms Local Outlier Factor, Random Forest and Term Frequency Inverse Document Frequency. To decide which one has the more efficiency to detect the anomaly.	In this research the work is only focused on the Anomaly detection algorithm s and their performa nce. But our research is about anomaly detection, classifica tion and fixetion.
Akshi Kumar Et. al. [9] Jul 2017	Classificatio n of Faults in Web Applications using Machine Learning	This research is to provide a model to detect fault and classify the fault and the fault's location using the machine learning tools and advanced signal procession.	Uses Text Mining and Machine Learning technique to classify the faults of three open source Web Application s, namely, the qaManager, bitWeaver and WebCalend er.	This research is about the power grids fault detection, where as our concepts is to detect the fault in running software using the ML.
Manik Lal Das Et. Al. [35] Nov 2020	Web Application Attack Detection using Deep learning	Using the deep learning, detection of web attacks, done through HTTP/HTTP S.	Auto encoder is used in this model to learn the sequences of word and wight each word or character. ECMLKDD datasets are used to train the system so	This research uses ML process to detection of web attacks and in our proposal we'll use the ML technique s to detect the

3. Problem Identification

3.1. Errors and Anomalies

Mistakes made during development, which can be more conceptual or logical, are the errors in software. Bugs are incorrect behavior or the exposure of these errors at the time of functioning of the software. Sometimes, the failure of software may happen due to configuration, improper data read, and locking of a resource for infinite time, network, and other hardware resources. There may be a chance of having 15 to 50 bugs in each 1000 lines of code. The more the requirements, the more the code, and the more the code the more the code the more the errors. Not only the codes but the more the requirements of automation increase, the more the complexity of configurations and infrastructures increase. Doing early detection of errors, rigorous testing is being done as a part of the SDLC process.

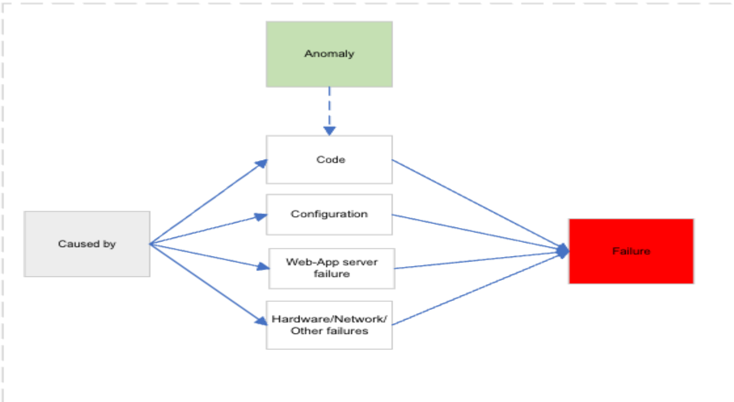

Fig 2. Anomalies that can appear in a web application at run time

3.2. Traditional Approach to Report Production Anomaly

Product industries have relied on customer feedback or reports to identify issues with their products. This means that if a user does not report an issue promptly or accurately enough, it may take weeks or months before the problem is identified and fixed by the company.

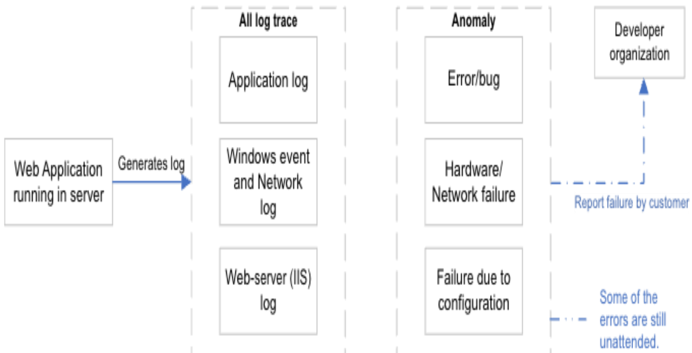

Fig 3. Traditional error/failure reported by customers/users

4. Proposed System

However, what if we have a system that is efficient enough to learn (Detect and Classify) from the errors/bugs/failures that occur in the production web application and Fix/notify the company immediately? Increasing reliability, consistency, and customer satisfaction will directly impact the quality.

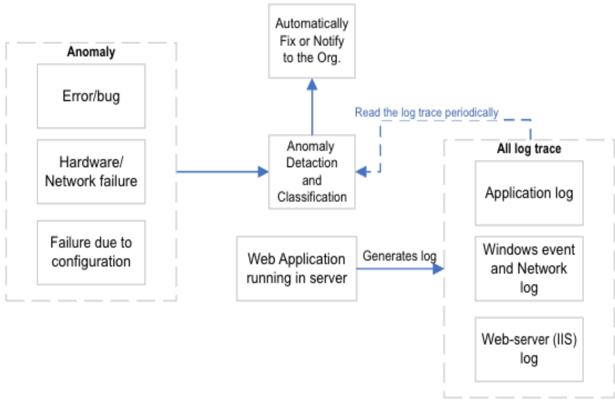

Fig 4. Flow Diagram of the proposed approach4

Overall, having such self-learning and self-correcting systems in place within the train/bus/plane automation industry can help increase safety, reduce downtime, and improve the overall reliability of the product.

Our research aims to develop a framework that can Detect, Classify, and Fix anomalies occurring in web application software (Web/App), which may arise from code, configuration, hardware, or network. We will use a machine learning algorithm to detect-classify and learn from the errors and fix them using an AI algorithm module on the fly. If the errors cannot be fixed, we will notify the developer company to address the problem. This will help to bridge the gap to error detection using ML and correction at run time for web applications.

In conclusion, with the increasing reliance on softwaredriven solutions across industries and sectors worldwide, ensuring high accuracy and reliability has become more important than ever before. By leveraging API-based monitoring tools & techniques along with other best practices like continuous testing & deployment methodologies, businesses can deliver top-notch quality services/products while minimizing risks associated with bugs/errors encountered during the development/deployment phases thereof.

5. Methodology

The goal of the study is to Design and implement a model that provides a framework to increase the productivity and reliability of an app/web application.

H1: Detection and Classification of anomalies occurring in the web applications will increase the efficiency of that web application (used by the customer).

Design a client and API and develop an algorithm to detect the error and other environmental factors that can learn the error/failure occurs with the cause of arising of issues at the run time of the web application. And other environmental variables at that point in time. Such as virtual memory, Network delay or timeout, File not found, deadlock in the Read-write of a file, etc.

Detect Anomaly Using Pattern Search (DAUPS) - Soft computing Machine Learning algorithms are to be used along with LOF to detect and classify the errors that occurred in the system:

- Detect Anomaly Using Pattern Search (DAUPS).
- KNN-Local Outlier Factor (LOF)

H2: The efficiency of the web application will be enhanced when a detected anomaly in a web application is fixed.

Automate and develop an algorithm to analyze the learned anomaly by using Generative AI techniques and define the probable fixes to fix the anomaly automatically at the product level.

While analyzing the learned data, the optimization and decision-making would be done using the following algorithms:

- Generative AI.

Then either the anomaly will be fixed and notified, or notify the anomaly to the developer organization to provide the fix.

5.1. Modules of the proposed system

5.1.1. The Anomaly Learning System (ALS) The anomalies (Error/Bug/Failure) that occur in the production web application server would be Detected and Classified by the Anomaly Learning System using ML.

The module would be trained with a supervised and semisupervised algorithm. So that the Anomaly Learning System (ALS) can classify the issue that appeared in the system if required, learn as a new record, and synchronize with the cloud centrally.

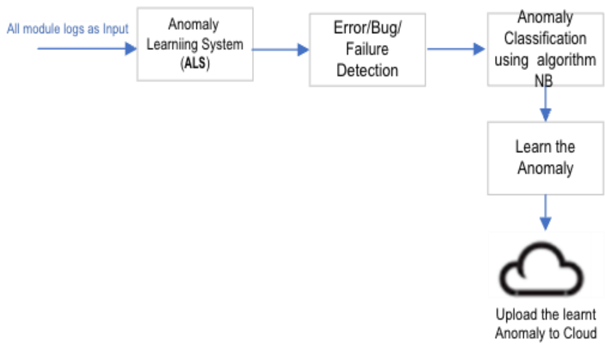

Fig 5. Anomaly Learning System (ALS)

5.1.2. Fix or Notify to organization with AI (FN) The responsibility of this module is to check for the best match of the solution and apply that. If no fix is found, then notify the development organization to fix that manually and update the knowledge base. The following process may be observed.

- If the FN cannot fix the issue using AI, then the Notify system will immediately raise concerns to the Owner company.

- If fixed, then update the AI knowledge base with the fix and notify the development organization. So that the fix can be used across the board for all web application servers.

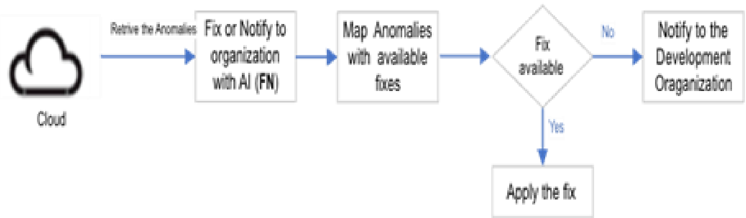

Fig 6. Fix/Notify to Organization (FN)

5.2. Characterizing the error/failure/anomaly

For the detection of anomalies in web applications, generally, the volume of logs generated by the usage of the application is taken into consideration. More usage of the application generates more logs. The more usage of the application, the more chances of increasing errors or anomalies. The modules involved in running a web application are:

- Web/App Application

- Web server (hosts the web application)

- Windows event log (manages hardware and network resources) The Anomaly includes:

- Code error – an error occurs in web application code.

- Server error – error/bug occurs in the web server while handling the request and response or any configuration error.

- Hardware/Network failure - failure occurs in server hardware, network, or in communication with the cloud resources.

5.3. Research Methodology:

The characteristics of both methods, Qualitative and Quantitative, i.e., the Mixed method, would be used here to validate the research. Exploratory research type, where we need to know elaborately how the components of the proposed system work:

- A client component would be deployed in the webserver to detect and classify the anomaly that occurs in the web application and server.

- An API will talk to the above client and update the central knowledgebase with the anomaly that is learned in the above step. Along with the learning a new event would be triggered to identify a fix and initiate that.

- Once an anomaly is detected that is not learned before, update the knowledge base. Initiate the process to match the fix for that anomaly. If found, trigger that to fix the anomaly. If that is resolved, map the fix with the anomaly. Else raise concern to the development organization.

- If the anomaly is already in the knowledgebase and the fix is mapped, then trigger that to fix the anomaly.

It is planned to split the activity into phases with the proper guidance which will cover all the following, that will take care of all anomalous events in depth.

The project will work in the following pattern:

- Gather logs.

- Detection of anomalous events from the log using soft computing ML algorithms.

- Classify the detected anomalies using soft computing ML algorithms.
- Learn the anomaly in a centralized knowledge base.
- Map, Apply, and Notify the fix using Generative AI. If not fix is not available, notify the development Organization.

5.3.1. Data Collection

ORACLE Ind. has multiple internal web applications hosted on its internal servers. We have chosen one of the web applications to collect our required logs. For the web application, there are different log streams, such as Application code logs, IIS logs, and Windows event logs. The application is developed on the .Net framework using C#, Python, and PowerShell for logging. Log4Net is used in code. The logs are being collected periodically (once every 5 days).

Fig 7. Application log trace

Fig 8. IIS HTTPERR log trace – web server

Fig 9. Inetpub log trace – web server

Making an effort to reproduce an error in production or chasing IIS logs or error logs is not so easy. The hidden point is, there are probably several errors going on that we aren't even aware of. The easy way to see an error is when a customer raises a concern by saying, the production site is throwing errors!

Collection information from the different log traces of the web application and the web servers (like IIS) is the most important point to detect the runtime error.

By browsing the internet, we can get the knowledge of how to identify the type of exceptions/errors and segregate them into groups. We may not encounter all types of exceptions or errors in a sample web application, and we can get more from the internet.

5.3.2. Process the gathered information and consolidate them

Information would be compiled and consolidated into volumes that are collected through the different sources. These volumes would be used at the time of doing experiments, formalization of statistical models, and the preparation of a thesis or dissertation.

An earlier collection of the immense volume of information would be an asset to the research work.

5.3.3. Search and collect the software packages and, if required, develop

No availability of such software packages has a direct impact on this line of research. Only one left-out point - we need to implement a prototype (collection of several scripts) to detect, classify, and fix such runtime errors.

5.3.4. Verifying the test results, identifying the findings of experiments, and confirming the research

Several verifications would be subjected to the generated outputs that would be traced from a web application. The difference in types would be noted down for further classification and fixing.

6. Dataset and Algorithms Discussed

6.1. Dataset

The real-time logs of a web application, web server, and Windows event are collected from a web application that communicates with the OCI cloud that is internal to ORACLE.

6.1.1. Application Architecture

The real-time application from which we have collected the log is split into three layers:

- Client side.
- Web server.
- Cloud resources.

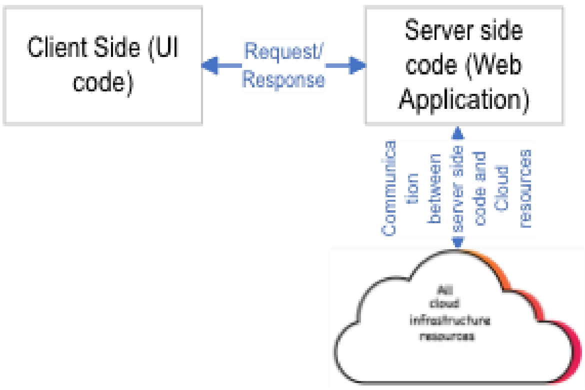

Fig 10. The architecture of Real-time Web-Application from where the log was collected

6.2. Algorithmic Concepts

Machine Learning:

Learnings that are done by a machine means the skill in the form of artificial intelligence that enables the computer to learn specific tasks without being programmed specifically. A piece of software program that uses multiple algorithms to learn about a targeted object.

Supervised Learning:

Based on past knowledge, Supervised learning requires labeled data. To identify, it applies the past knowledge to the new data.

Unsupervised learning:

Whereas the unsupervised draws the conclusion from the datasets directly. The unlabelled data are used in this learning.

Logs as Input:

Web applications or any software applications generate large data system logs. They are typically unstructured text written line by line in a file by maintaining the time sequence. While the log line is being written by a code statement it contains a Constant and a Variable part. The constant part is directly Written by the coder, but the variable part is written by the system with that specific time value. It may be data or maybe an error exception. The log format depends on the developer, where the logger may be in-house or logger utilities software available in the market. For example, Log4Net is for the .Net platform, and Log4J is for Java platform development.

Fig 11. Sample of a log which is taken as an input

Anomaly:

Something that is not normal surrounding it, i.e., the thing that is unusual or irrelevant or unacceptable called an anomaly. This came from the Greek word "anomolia".

6.3. Detect Anomaly Using Pattern Search (DAUPS):

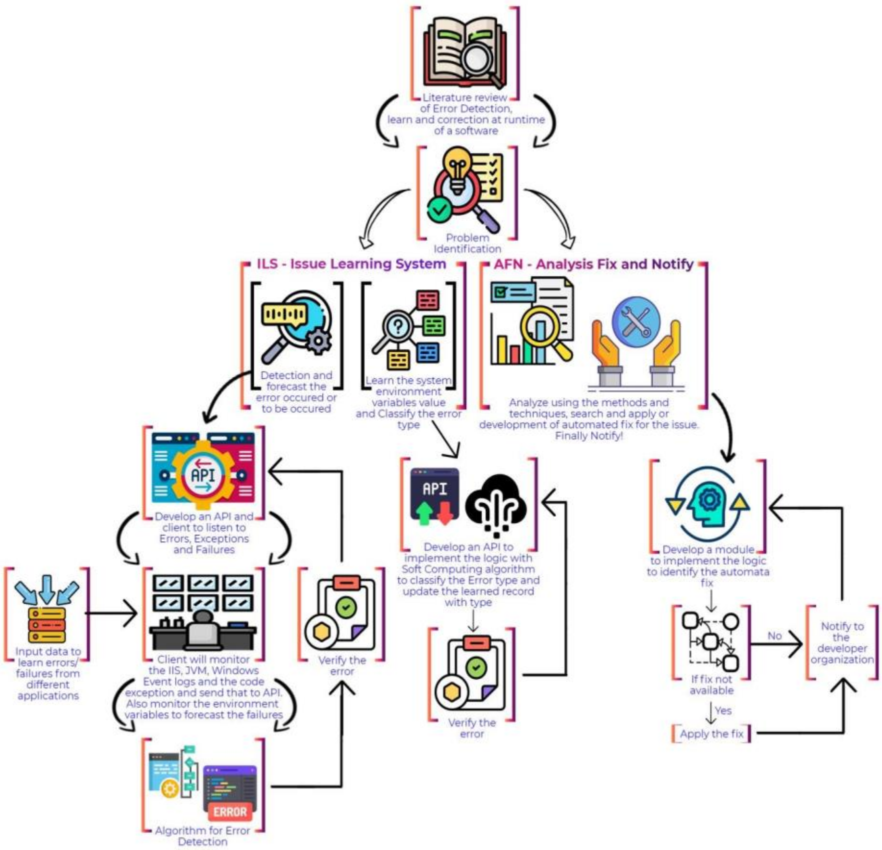

Fig 12. Flowchart of the overall Research Outline

This approach to detecting anomalies in a log file is based on the given pattern initially. Once the pattern is identified in the log string, then read the whole string or context of the log stream. Check the occurrence of the exception/error with the knowledgebase. If this is already in the knowledgebase, then increase the occurrence against that log stream; if it does not exist in the knowledgebase, then enter this with the frequency of occurrence.

6.4. KNN-Local Outlier Factor (LOF)

A data point that is far from the rest of the data points is called an Outlier. A point is treated as an outlier based on its local neighbors called a local outlier.

The algorithm's local outlier factor works based on the local density. Internally, this uses the KNN (K-Nearest Neighbours) algorithm, where K is the number of data points to be referenced.

- Calculate the Reachability Distance
- Calculate the local reachability Density of all K neighbors.

The value with the high density is the local outlier.

7. Results and Discussions

7.1. Error Detection that occurred in Module 1

7.1.1. Introduction & the EDO system

The error detection system is such type of system where that tries to read the exception/error that occurred in the application, webserver, and Windows logs.

- This module will take error inputs from several log sources (Application log, IIS server log, IIS client log, Windows system log, etc.)
- Building a keyword patterns library of exception/error/failure/warning will help us to detect/forecast the error.
- The responsibility of this module is to filter out the exceptions/errors/failures/warnings from the log traces.
- The initial detection of the errors will be treated as supervised learning.

7.1.2. Learn the system environment variable values and classify the error types We have the following error types:

- Configuration error
- Resource error
- Other failures (Hardware, Network, etc.)

7.1.3. The proposed detection and classification

algorithm DAUSP:

- This is a supervised ML model to detect the errors that occur in the logs.
- Let 'P' be the set of predefined pattern key word or data to identify the error and exceptions in log-trace. P = (p1, p2, …, pn).
- If we have a 'T' number of tuples (log traces), where 'T' = (t1, t2, …, tn). For each pattern, 'p' would be checked for the match with each tuple

't'.

- Now, all matched tuples are kept in a list 'L' and will be used as a 'T' in LOF to classify them.

Note: We would be using the list of negative adjectives (how the sentence sounds positive/negative) pattern to identify and learn the errors.

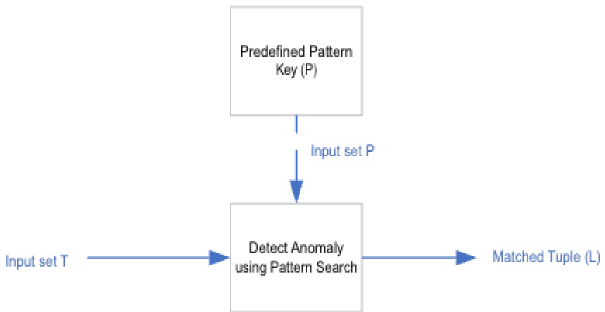

Fig 13. Detect Anomaly using Pattern Search.

KNN-LOF:

KNN (K-Nearest Neighbours)

LOF is an unsupervised Anomaly detection algorithm that internally uses the KNN algorithm to detect the distance of the selected data points with K-th data points.

And in LOF the calculated distance in KNN is being used to calculate the density.

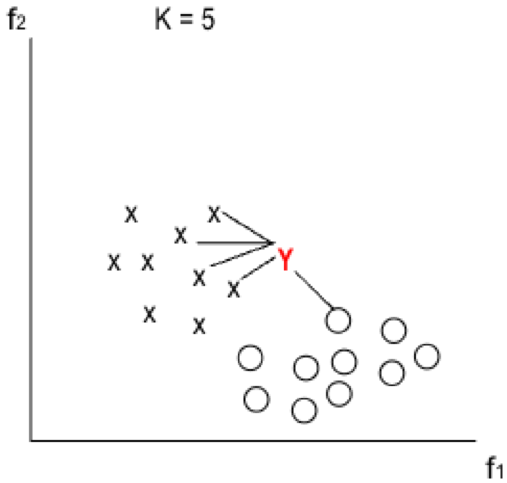

Fig 14. Classified the datapoint (f1 and f2 are the two categories here) Y based on the distance between the new data points using K = 5.

K-Value is a hyper parameter that gives us the count by using which we check the K (number of) nearest data points.

Euclidian distance for classification: Between the two data points (x1,y1) and (x2,y2), we can find the distance by using the formula: √(x2-x1)2+(y2-y1)2. Here, based on the K value, we can identify the new data point, which means we can classify the error that is detected in an earlier stage.

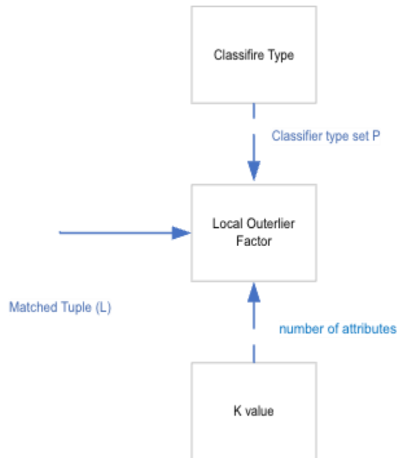

Fig 15. KNN with the input from earlier stages and Classifier type

Suppose we have a classifier type:

- File not found may have attributes File Location, File Name, Failed action, etc.
- Resource Locked by another process may have attributes File Location, File Name, Failed action, Locked by, etc..

For now we are not directly applying the Local Outer Factor (LOF) in classifying the errors. If the model fails to classify the errors using KNN, then LOF will be implemented.

7.2. Error Correction that occurred in Module 2

7.2.1. Analysis and Notify

A model to analyze and fix it comes into the picture once the error is detected and classified.

This model has three sections:

- The different types of anomaly are to be fixed in different timeframes and environments.

o Run-Time Code error is to be fixed in a parallel environment as triage of fixing the code may create downtime in a production environment.

o Run-time configuration errors can be fixed directly in a production environment by keeping a backup of the original configuration.

o Web-Server error can be fixed at the run time in a production environment.

o Hardware/resource / Networking error can be fixed at run time in a production environment.

- Identify the suitable fix available from knowledgebase.
- For different types of anomaly, the way of applying a fix would be different.

The Generative-AI model would be used to apply the fix. In our case, we will be using the text-to-text model to fix the anomalies.

Conclusion

The studies on proactively addressing runtime anomalies in web applications discovered that applying sophisticated machine learning approaches has much potential to improve the overall quality, efficiency, and security of Internet-based services. Through its accurate use of complex models such as deep learning, anomaly detection algorithms, and reinforcement learning, it is possible to detect and curb misbehavior and system instability before they become problematic. The approach is also useful in the real-time identification of anomalies and is simultaneously used in predictive maintenance since it can predict possible defects that may affect users. Further, when the machine learning models are integrated with the current monitoring tools, it offers the ability to have an end-to-end check using a systematic approach for managing irritations. In general, the use of modern machine learning technologies in this field can be considered as the shift to a new level of Web applications development, which will be more intelligent, adaptive, and robust in terms of protection and management, and the perspective for further developments in the field of automated control and security systems.

References

[1] Valentina Lenarduzzi, Alexandru Christian Stan, Davide Taibi, Davide Tosi. “A Dynamical Quality Model to Continuously Monitor Software Maintenance.” https://www.researchgate.net/publication/319187942_A_D ynamical_Quality_Model_to_Continuously_Monitor_Soft ware_Maintenance

[2] Ruchika Malhotra. “A systematic review of machine learning techniques for software fault prediction.” https://www.sciencedirect.com/science/article/abs/pii/S1568494614005857

[3] Awni Hammouri, Mustafa Hammad, Mohammad M Alnabhan. “Software Bug Prediction using Machine Learning Approach.”https://www.researchgate.net/profile/Mustafa-Hammad2/publication/323536716_Software_Bug_Prediction_using _Machine_Learning_Approach/links/5c17cdec92851c39eb f51720/Software-Bug-Prediction-using-Machine-LearningApproach.pdf

[4] Matthew Arnold, Martin T. Vechev, Eran Yahav. “QVM: an efficient runtime for detecting defects in deployed systems.” https://www.researchgate.net/publication/221320711_QV M_An_Efficient_Runtime_for_Detecting_Defects_in_Dep loyed_Systems

[5] Syahana Nur’Ain Saharudin, Koh Tieng Wei and Kew Si Na. “Machine Learning Techniques for Software Bug

Prediction: A Systematic Review.” https://thescipub.com/pdf/jcssp.2020.1558.1569.pdf

[6] Chang Lou, Peng Huang, Scott Smith. “Comprehensive and efficient runtime checking in system software through watchdogs.” https://changlousys.github.io/paper/watchdog-hotos19preprint.pdf

[7] Sana Ullah Jan, Young Doo Lee, In Soo Koo. “A distributed sensor-fault detection and diagnosis framework using machine learning.” https://www.sciencedirect.com/science/article/abs/pii/S002 0025520308422

[8] L. Seabra Lopes and L.M. Camarinha-Matos A machine learning approach to error detection and recovery in assembly https://www.academia.edu/33548490/A_machine_learning _approach_to_error_detection_and_recovery_in_assembly

[9] Classification of Faults in Web Applications using Machine Learning Akshi Kumar, Rajat Chugh, Rishab

Girdhar, Simran Aggarwal https://www.researchgate.net/publication/317031520_Clas sification_of_Faults_in_Web_Applications_using_Machin e_Learning

[10] Web Application Attacks Detection Using Machine Learning Techniques Rodrigo Martínez, Gustavo Betarte,

Alvaro Pardo https://www.researchgate.net/publication/329514915_Web _Application_Attacks_Detection_Using_Machine_ Learnin g_Techniques

[11] Software fault prediction using error probabilities and machine learning approaches. Karuppusamy, S https://shodhganga.inflibnet.ac.in/handle/10603/341273

[12] A robust semi-supervised SVM via ensemble learning Dan Zhang, Licheng Jiao, Xue Bai, Shuang Ru Wang https://www.researchgate.net/publication/322998378_A_R obust_Semi-Supervised_SVM_via_Ensemble_Learning

[13] Machine Learning Techniques for Intrusion Detection: A Comparative Analysis Yasir Hamid, Sugumaran Muthukumarasamy, Ludovic Journaux https://www.researchgate.net/publication/309638541_Mac hine_Learning_Techniques_for_Intrusion_Detection_A_C omparative_Analysis

[14] Software Defect Prediction Analysis Using Machine Learning Techniques Aimen Khalid, Gran Badshah, Nasir

Ayub, Muhammad Shiraz, Mohamed Ghouse https://www.mdpi.com/2071-1050/15/6/5517

[15] Software Fault Prediction Using Deep Learning Techniques Iqra Batool, Tamim Ahmed Khan https://assets.researchsquare.com/files/rs 2089478/v1_covered.pdf?c=1664894193

[16] A Real-Time Detection Method of Software Configuration Errors Based on Fine-Grained Configuration Item Types Li Zhang, Shengang Hao and Meng Ming https://www.hindawi.com/journals/sp/2022/4415366/

[17] A Novel Machine Learning Approach for Bug Prediction Shruthi Puranik, Pranav Deshpande, K. Chandrasekaran https://www.sciencedirect.com/science/article/pii/S1877050916315174

[18] Analysis on Detecting a Bug in a Software using Machine Learning Rashmi P, Prashanth Kambli https://www.ijrte.org/wp-content/uploads/papers/v9i2/B4119079220.pdf

[19] Bug Prediction with Machine LearningmGustav Rehnholm, Felix Rysjö https://www.diva-portal.org/smash/get/diva2:1563558/FULLTEXT01.pdf

[20] Machine learning techniques for web intrusion detection— A comparison Truong Son Pham, Tuan Hao Hoang, Van

Canh Vu https://www.researchgate.net/publication/311314204_Mac hine_learning_techniques_for_web_intrusion_detec tion_-_A_comparison

[21] An optimized machine learning approach for fault detection and reliability estimation of software testing Sudharson, D http://hdl.handle.net/10603/342415

[22] HYBRID RELIABLE AND SECURED PREDICTION MODEL FOR SELF HEALING SYSTEMS Dr.S.P. Rajagopalan http://hdl.handle.net/10603/118192

[23] Fault free software engineering framework to detect errors and lead to better software development phase.Rajkumar N http://hdl.handle.net/10603/261939

[24] Expert system for Software Error Detection and Correction SEDC in Integrated Development Environment IDE Josephine, MShttp://hdl.handle.net/10603/265095

[25] Towards Reliable AI Applications via Algorithm-Based Fault Tolerance on NVDLAMustafa Tarik Sanic, Cong Guo, Jingwen Leng, Minyi Guo, Weiyin Ma https://ieeexplore.ieee.org/abstract/document/10076581

[26] Efficient Software-Implemented HW Fault Tolerance for TinyML Inference in Safety-critical Applications Uzair Sharif, Daniel Mueller-Gritschneder, Rafael Stahl, Ulf Schlichtmann https://ieeexplore.ieee.org/abstract/document/10137207

[27] Automata based software reliability model Wason, Ritika http://hdl.handle.net/10603/88404

[28] Machine learning-based run-time anomaly detection in software systems: An industrial evaluation Fabian Huch, Mojdeh Golagha, A. Petrovska, Alexander Krauss https://ieeexplore.ieee.org/document/8368453

[29] Leveraging Machine Learning to Improve Software Reliability Song Wang https://core.ac.uk/download/pdf/169432113.pdf

[30] Software Defect Detection Using Machine Learning Techniques Jaswitha Abbineni; Ooha Thalluri https://ieeexplore.ieee.org/document/8553830

[31] Anomaly Detection in Log Files Using Machine Learning Techniques Lakshmi Geethanjali Mandagondi https://www.diva-portal.org/smash/get/diva2:1534187/FULLTEXT02.pdf

[32] Machine Learning-Based Run-Time Anomaly Detection in Software Systems: An Industrial Evaluation Machine Learning-Based Run-Time Anomaly Detection in Software SystemsFabian Huch; Mojdeh Golagha; Ana Petrovska;

Alexander Krauss https://www.researchgate.net/publication/325492786_Mac hine_learning-based_run-time_anomaly_detection_in_software_systems_An_industr ial_evaluation

[33] Literature Reviews Knox College Library https://library.knox.edu/friendly.php?s=literaturereview#:~:text=A%20literature%20review%20is%20the,fo r%20a%20primary%20research%20project.

[34] Experiences from using snowballing and database searches in systematic literature studies Deepika Badampudi; Claes Wohlin;Kai Petersen https://dl.acm.org/doi/10.1145/2745802.2745818

[35] Web Application Attack Detection using Deep learning Manik Lal Das https://arxiv.org/abs/2011.03181

[36] Implementation of Fault Detection Framework For Healthcare Monitoring System Using IoT, Sensors In Wireless Environment Dr. K. P. Paradeshi https://www.researchgate.net/profile/KutubuddinKazi/publication/363801133_Implementation_of_Fault_D etection_Framework_For_Healthcare_Monitoring_System _Using_IoT_Sensors_In_Wireless_Environment/links/632 ee4f486b22d3db4dbd8b0/Implementation-of-FaultDetection-Framework-For-Healthcare-Monitoring-System- Using-IoT-Sensors-In-Wireless-Environment.pdf

[37] Web Statistics https://medium.com/@aplextorlab/webstatistics-69493eebbd01

[38] Generalized software fault detection and correction modeling framework through imperfect debugging, error generation and change point Iqra Saraf and Javaid Iqbal https://link.springer.com/article/10.1007/s41870-019-00321-x

[39] A model of software fault detection and correction processes considering heterogeneous faults Ruijin Xie, Hui Qiu, Qingqing Zhai, Rui Peng https://www.researchgate.net/publication/362691735_A_m odel_of_software_fault_detection_and_correction_process es_considering_heterogeneous_faults