Migration Letters

Volume: 21, No: S1 (2024), pp. 1082-1089

ISSN: 1741-8984 (Print) ISSN: 1741-8992 (Online)
www.migrationletters.com

Machine Learning Based Approach On Full Stack Development Utilizing Mongodb To Diagnose Faults In Complex Mechatronics System

Rajesh Sharma R[1], Ketema Adere Gemeda[1], Mesfin Abebe[1], Subramanian Selvakumar[1]
Akey Sungheetha[2], Chinnaiyan R[2], Sengottaiyan N[3]

Abstract.

In the context of computerization and industrial plants, the concept addresses the memory safeguarding overhead and the significance of creating a diagnosis component for complex mechatronic systems. In order to minimize needless repairs and cut expenses, the module seeks to detect errors (Fault De- tection) and isolate their underlying causes (Fault Isolation). FastAPI Python, Docker, which is MongoDB, as well as machine learning algorithms make up the tech stack used, and AWS S3, AWS EC2, along with AWS ECR are neededfor the infrastructure. MongoDB is shown in the project architecture, and Com- pass needs to be set up for data storage. For the purpose of improving system performance, productivity, and safety, the evaluation section is essential. Time series readings from sensors make up the dataset utilized for fault detection, where SensorID stands in for a temperature detector in a manufacturing envi- ronment. The goal of the issue of binary classification is to determine if a par- ticular part of the heavy-duty vehicle's Air Stress System (ASS), which uses compressed air stress for braking, was the cause of the failure.

Keywords: *Fault detection, fault isolation, Python, MongoDB, machine learn-ing, data storage, time series, sensors.*

Introduction

1 Development of Full Stack Development and Its Significancein Contemporary Web Applications

A full stack d[1]eveloper's "classic" skill set consists of the following HTML, CSS, and JavaScript; - one or more well-known web frameworks, like Vue, React, or An- gular. It includes databases and version control systems (at least Git). It also brings in the understanding of best practices for web, visual, and user experience design. The security best practices and obstacles related to web security are incorporated along with web server configuration, installation, and analytics. The programming lan-

[2,3] Department of Computer Science and Engineering, Center for Electrical System and Electronics, Adama Science and Technology University, Adama, Ethiopia.

[4] Department of Electrical and Computer Engineering, Bahir Dar Institute of Technology, Bahir Dar University , Bahir Dar, Ethiopia.
[1,5,6,7] Department of Computer Science and Engineering, Alliance University, Bangalore – INDIA

guages, such as Python, PHP, and Ruby (and, more recently, Go and Scala) are fre- quently used in web backend development [1].

The term "full stack" describes an understanding of all computer applications, in- cluding system design, front-end UI creation, REST API creation for endpoints, and database-driven back-end data handling [2].

The LAMP (Linux, Apache, MySQL, PHP/Perl) stack and Java (Java EE, Spring), which consists of various programming languages, served as the foundation for web development initially. This multi-language paradigm was resolved by JavaScript, which is the foundation of the MEAN (MongoDB, Express, AngularJS, Node.js) stack. Since the initial application was created primarily to demonstrate how the MEAN stack was implemented, the report concentrates more on how the MEAN stack's components were implemented in an application than it does on the applicationitself [3].

Managing both the construction and operations phases is known as full stack engi- neering (DevOps). For the development portion, moderate expertise in programmingis needed, and for the operations field, administrative knowledge is required. A few coronavirus statistics will be connected to the data that was scraped. These servicesare essential to achieving the backend web scraping function's requirement to operate at a specific time interval.

2 Improving Fault Finding and Isolation in the ComplexMechatronic Systems: An Overview

The technology that solves the data management issues faced by many modern or- ganizations is called client/server computing [4]. In order to quickly prototype solu- tions for Internet of Things and smart products, we attempt to determine the scientific and architectural needs of a symmetrical stack by abstracting the problem from actual implementations [5]. We look into a challenging stacking issue that arises from inte- grated steel production's storage planning for steel slabs. These stacking tasks have theoretical appeal in addition to their practical significance. We demonstrate that our stacking challenge is already PSPACE-complete in its most basic form [6]. We use qudits, which provide a natural platform for such non-binary problems, and the quan- tum approximate optimization algorithm (QAOA) to explore the correlation clusteringproblem [7]. Not only can a quantum computer solve pertinent problems that are be- yond the capabilities of current classical computers, but it also heralds a paradigm shift in computing practices [8]. Different methods and strategies are employed by different approaches to address the mapping problem. They also differ in the expendi-ture of function that is optimized during the mapping process and serves as the map- per's performance metric. The standard metrics include the number of SWAPs for gate overhead, the number of time-stamps for circuit depth as well as latency over- head, and the probability of failure or success rate [9]. There are several challenges associated with using these pen-and-paper-based medical records, including financial, technological, knowledge, human, and legal difficulties. An electronic medical rec- ords framework is put forth to address this issue. With this framework, the conven- tional issues with pen-and-paper systems are resolved [10].

Experimental results on several public MR datasets confirm that full-stack learning(FSL) outperforms current state-of-the-art techniques in every task. FSL offers a lot ofpotential to improve the useful workflow of MRI in radiotherapy and medical diagno- sis [11]. TravisTorrent is unique in that every one of its 2,640,825 Travis constructs iscombined with dynamically combined project data extracted through GHTorrent,textual build log analysis results, a reference to the GitHub commit that started the build, and meta data from Travis CI's API [12]. But we think this amount of time is reasonable because verification yields significant assurances of correctness and can beperformed once per release cycle [14]. In order to comprehend the difficulties and potential in creating smart contracts on blockchains, one of the findings indicates that there exists no practical means of ensuring smart contract code security. The authors provided a thorough examination of the security concerns with smart contracts, but they did not provide any experimental confirmation of actual cases [13].

A highly customized flowing hardware architecture that uses FPGAs to accelerate CNNs fully, with the goal of increasing compute efficacy for streaming applications. In order to facilitate the inference of mainstream CNNs with various topologies, the suggested accelerator maps the majority of computational functions, i.e., convolution- al and deconvolutional layers, into a single, unified module and sets up the residual aswell as concatenative relationships between the features with high efficiency[14].

We have implemented CNNs with powers-of-two weights using an effective Selec-tor-Accumulator (SAC) architecture, which is a standout feature of our approach and helps us achieve high energy efficiency. For a conventional 8-bit MAC, SAC signifi- cantly lowers needed hardware capacity (4.85x fewer searches tables) and electrical consumption (2.48x) as opposed to an FPGA implementation[15].

Nonetheless, the majority of research to date has been devoted to investigating physical layer solutions and comprehending the science of terahertz devices, circuits, and propagation. The effective design of the entire communication stack is necessary for the integration of this technology into complicated mobile networks, though, in order to solve system- and link-level issues with network configuration, management, coordination, conservation of energy, and end-to-end connectivity[16].

3 Creating a Fault Forecasting Algorithm for Industrial Plant Operating Conditions

In order to evaluate the industrial plant's operating conditions, we created and putin place a fault forecast algorithm that takes advantage of sensor data fusion on top of the suggested infrastructure. The accuracy results from testing the model we suggest- ed are displayed in the result section along with a comparison with a conventional deep neural network (DNN) structure [17].

4 A Comparison Using a Traditional Architecture

The current industrial fashion toward automation and industrial plants is pushing us toward increasingly complex mechatronic systems operating in an unpredictable, evolving environment. The development of a diagnosis module is imperative in order to identify potential faults affecting the functioning of these systems (Fault Detection)and identify the root causes of those faults (Fault Isolation).

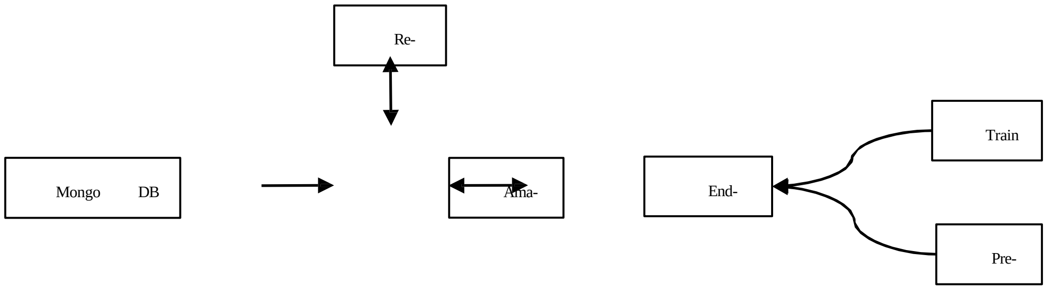

Fig 1. MongoDB based architecture

The issue is cutting costs because of pointless repairs. Thus, reducing the number of incorrect forecasts is necessary for the data collection as shown in Fig 1.

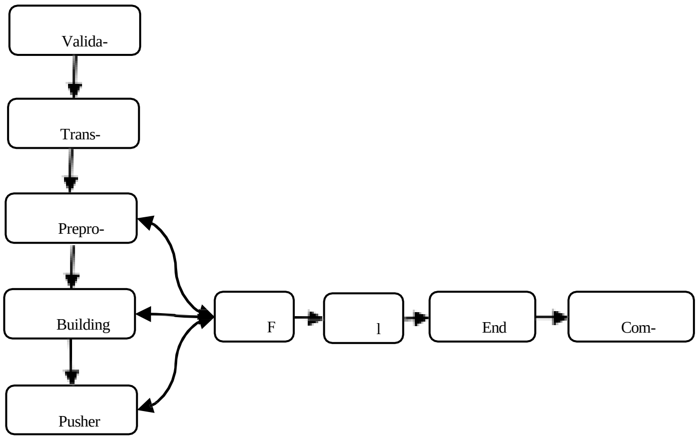

Figure 2. Fault detection development stages

Tech Stack Employed FastAPI Python, Docker, MongoDB, and machine learning algorithms. Infrastructure is needed. Get Actions for AWS S3, AWS EC2, and AWS ECR Terraforma for project architecture as shown in Fig 2.

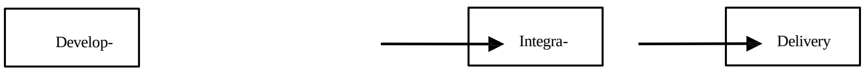

Fig 3. Full stack deployment

Since we will be employing MongoDB for data storage, please ensure that Mon- goDB is installed on your local system and that Compass is working before we begin the project. In order to access services you also need an AWS account for deploymentarchitecture as shown in Fig 3.

5 Perform Sensor Error Identification for diagnosing faults

When we fix the link circumstance to every scenario, the average from beginningto end throughput is approximately 48.1 Mbps, and an signal-to-interference-plus- noise ratio (SINR) $\geq$ 30 dB (Fig. 2) leads to Signal, S > Ratio,R. The average from beginning to end throughput is 47.1 Mbps, 47.8 Mbps, 48.1 Mbps, as well as 47.5 Mbps. The SINR situations is sufficiently high to allow for S > R. When comparing the throughput scenarios, there is a small decrease because in one or two simulation operates, the SINR is inadequate, which results in S < R [18].

Consequently, in order to increase system productivity and performance and re- duce the potential for catastrophic failures that could endanger human life and proper- ty, a diagnosis section is required as presented in Table 1.

Table 1. Timestamp sensor values

Timestamp	Value measured on sensor at timestamp
2017-03-01T23:20:00+03:00	18.4798069
2017-03-02T04:00:00+03:00	19.53911209
2017-03-23T06:25:00+03:00	19.25019836
2017-03-23T19:35:00+03:00	18.96128464
2017-04-04T15:10:00+03:00	25.32162285

Time series of sensor measurements on individual sensors designated from a Sen- sor Id. The sensor is either disconnected or fails during the series of measurements. The memory protection overhead is specifically calculated as follows: $M = \hat{T}s - Ts$, (1), where $\hat{T}s$ and Ts are the scan times, respectively, both for and without memory- safe compilation. We give a thorough explanation of modeling Ts in our previous work [14]. ChIC-seq assays for small cell numbers were conducted similarly to our earlier study. In an instant cells were fixed in 1% formaldehyde-containing medium for ten minutes. The addition of 0.125 M glycine halted the reaction. After being per- meabilized for five minutes at room temperature using 700 µl buffer, 200 000 fixed cells were rinsed twice with 600 µl binding buffer ten millimeters Tris–Cl, 1 mM

EDTA, 150 mM sodium chloride (N 0.1% TX-100) and then resuspended in 250 µl binding buffer [23]. It is necessary to compute and check for obstruction every path inorder to calculate all possible thoughts between a given pair of nodes. The direct ray, or segment (RX, TX), is the first to be tested when taking increasing reflection orders into account. Then, all first order reflections—that is, the rays that originate at the TX, reflect off a triangle Ti, where i = 1,...,T, and arrive at the RX—are computed. Next, second order reflections begin at the TX and proceed up to a maximum contemplation order R by reflecting first on triangle Ti1, i1 = 1,...,T, then reflecting on triangle Ti2, i2 = 1,...,T, i2 = i1 [15].

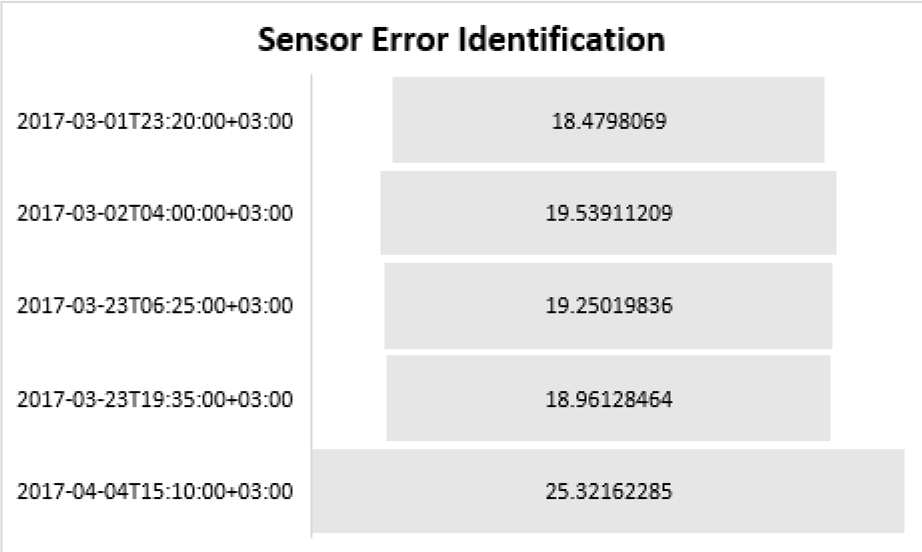

Fig 4. Fault diagnosis

The purpose of this dataset is to support the analytical component for fault detec- tion. SensorID = 1: A temperature sensor from the PT100 in an industrial setting with vibrations and dust as presented in Fig 4. An essential part of a heavy-duty automo- bile is the Air Stress System (ASS), which applies compressed air pressure to a piston to force the brake pads, to slow down the vehicle. Using an APS over a hydraulic system has advantages in that natural air is readily available and sustainable over the long term. This is an issue of binary classification, where the negative class denotes that the problem was triggered by something else, and the affirmative class suggests a specific APS component was the cause of the failure. ASS which produces pressur- ized air for use in braking as well as gear changes among other truck operations, is thesystem under investigation in this project. The positive class of the dataset represents component breakdowns for a particular APS system component. Vehicles in the nega- tive class are those that have malfunctioning parts unrelated to the APS system.

6 Conclusion

In conclusion, the prevailing industrialization and automation trend demands the development of complex mechatronic systems that can function in erratic and dynam-ic environments. The creation of a diagnosis component becomes essential to guaran- teeing the efficient operation of these systems. This module's functions include accu- rately identifying the faults' underlying causes (Fault Isolation) and detecting possible faults inside the systems (Fault Detection). We can increase industrial plants' depend- ability and efficiency, which will increase output and decrease downtime, by success- fully tackling these issues. To ensure that complex mechatronic systems continue to function properly in the face of changing industrial demands, a reliable diagnosis module must be developed and integrated.

7 Future Work

Creating sophisticated methods and algorithms for fault isolation and detection in intricate mechatronic systems. This might entail boosting the diagnosis module's pre- cision and effectiveness by applying machine learning and intelligence-based tech- niques. The diagnosis module can be integrated with the control system and other mechatronic system components to allow for automatic changes in response to faults that are detected and real-time monitoring. Creating appropriate interfaces and proto- cols for communication between the various modules would be necessary for this. extensive validation and testing of a diagnosis module in actual industrial settings. To guarantee the dependability and efficacy of the developed module, this would entail working with industry collaborators and obtaining data from various kinds of mecha- tronic systems.

References

1. Taivalsaari, Antero, et al. "Full stack is not what it used to be." Inter-national Conference on Web Engineering. Cham: Springer Interna- tional Publishing, 2021.
2. Heikkinen, Eetu. "Full Stack Development." (2022).
3. Adhikari, Anuj. "Full stack javascript: Web application developmentwith mean." (2016).
4. Vainikka, Joel. "Full-stack web development using Django RESTframework and React." (2018).
5. Mazzei, Daniele, et al. "A full stack for quick prototyping of IoT solu-tions." Annals of Telecommunications 73 (2018): 439-449.
6. König, Felix G., et al. "Solutions to real-world instances of PSPACE- complete stacking." European Symposium on Algorithms. Berlin, Hei-delberg: Springer Berlin Heidelberg, 2007.
7. Weggemans, Jordi R., et al. "Solving correlation clustering with QAOAand a Rydberg qudit system: a full-stack approach." Quantum 6 (2022): 687.
8. Bandic, Medina, Sebastian Feld, and Carmen G. Almudever. "Full- stack quantum computing systems in the NISQ era: algorithm-drivenand hardware-aware compilation techniques." 2022 Design, Automa-tion & Test in Europe Conference & Exhibition (DATE). IEEE, 2022.
9. Mamatha, G. S., and S. Shashank. "A Novel Approach for the Enter- prise Hospital Management Application with Full Stack Develop- ment." International Conference on Information and CommunicationTechnology for Competitive Strategies. Singapore: Springer NatureSingapore, 2022.
10. Shropshire, Jordan, Jeffrey P. Landry, and Steven S. Presley. "To-wards a consensus definition of full-stack development." (2018).
11. Wang, Zhiwen, et al. "Promoting fast MR imaging pipeline by full-stackAI." Iscience 27.1 (2024).
12. Beller, Moritz, Georgios Gousios, and Andy Zaidman. "Travistorrent: Synthesizing travis ci and github for full-stack research on continuous integration." 2017 IEEE/ACM 14th International

Conference on MiningSoftware Repositories (MSR). IEEE, 2017.

13. Chen, Hongsong, et al. "Security challenges and defense approachesfor blockchain-based services from a full-stack architecture perspec- tive." Blockchain: Research and Applications (2023): 100135.
14. Chekole, Eyasu Getahun, et al. "Enforcing full-stack memory-safety incyber-physical systems." Engineering Secure Software and Systems:10th International Symposium, ESSoS 2018, Paris, France, June 26-27, 2018, Proceedings 10. Springer International Publishing, 2018.
15. Lecci, Mattia, et al. "Accuracy versus complexity for mmWave ray- tracing: A full stack perspective." IEEE Transactions on Wireless Communications 20.12 (2021): 7826-7841.
16. Cao, Yaqiang, et al. "cLoops2: a full-stack comprehensive analytical tool for chromatin interactions." Nucleic Acids Research 50.1 (2022):57-71.
17. De Vita, Fabrizio, Dario Bruneo, and Sajal K. Das. "On the use of a fullstack hardware/software infrastructure for sensor data fusion and faultprediction in industry 4.0." Pattern Recognition Letters 138 (2020): 30-37.
18. Poddar, Hitesh, et al. "Full-Stack End-To-End mmWave SimulationsUsing 3GPP and NYUSIM Channel Model in ns-3." arXiv preprint arXiv:2302.12385 (2023).
19. Metrôlho, José Carlos, et al. "Prepare Students for Software Industry:A case study on an agile full stack project." The Seventeenth Interna- tional Conference on Software Engineering Advances. IARIA, 2022.