Causal Inference Framework for Root Cause Analysis in Ci/Cd Pipeline Failure Using Agentic Test Automation

Yashvardhan Rathi*[1], Savi Grover[2]
[1]Data Platform Engineer, Truist Financial Services, Atlanta, GA, USA
[2]Independent Researcher, Software Quality Engineer, Rahway, New Jersey, USA
Rathi.yashvar@gmail.com

Abstract. The process of Continuous Integration and Continuous Deployment (CI/CD) has become even more complex and risky, and the failure analysis steps have become dependent on the use of correlation-based approaches, which tend to correlate symptoms instead of looking into the root cause of the problem. In this paper, the author has come up with the concept of a causal inference framework for root cause analysis of CI/CD pipeline failures, devised through the use of agentic test automation techniques, finding the actual causal chains that exist beyond considering infrastructure changes, code commits, and testing environment configurations. By incorporating the use of self-driven AI agents within the testing process, the proposed framework strips away the involvement of the confounding variables, establishes the causal dependency, and provides concrete remediation recommendations. Apart from enhancing the efficiency of the process within the CI/CD pipeline, these steps have incorporated the necessary cybersecurity measures the value in the form of the detection of malicious/anomalous interventions, along with the use of blockchainbased audit trails, which enhance the transparency, trust, and accountability within the process of causal evidence. Finally, the proposed process has given special emphasis to the aspect of human-computer interaction in the form of explainable causal reasoning, beneficial visualization, and special emphasis for the developers/operators. The future perspective regarding the process has outlined the extension of the causal inference agents within the cross-organization DevSecOps domain, thus creating the basis of ethical, transparent, and benchmarked AI-driven AI governance within the software delivery process, thereby making the process of causal inference the next paradigm shift within the context of next-generation resilient, automated, and secure CI/CD pipeline process.

Keywords: Causal inference, Root cause analysis, CI/CD pipeline failures, Agentic test automation, Cybersecurity safeguards, Blockchain auditability, Ethical AI governance

1 Introduction

Background The Continuous Integration and Continuous Deployment (CI/CD) pipelines have became the spine of modern are now the backbone of contemporary

S. P. Vijayaragavan et al. (eds.), *Proceedings of the Global Conference on Sustainable Energy Systems, Smart Electronics and Intelligent Computing (GCSESEIC 2025)*, Advances in Engineering Research 297,
https://doi.org/10.2991/978-94-6239-654-8_69

software development methodologies, enabling fast practices, facilitating the rapid and efficient delivery of software solutions at scale. The automation of build and deploy processes using the deployment tasks through CI/CD pipelines reduces the minimizes human interaction and makes the system more malleable, ensuring the adaptability of the systems to meet varying business needs [1]. Recently, with the adoption trend, with the growing use of microservices architecture, containerization, and cloud-native applications, the complexity of the CI/CD pipelines has multiple many fold increased multifold. Although complexity enables innovation, it has remained an impediment in identifying and dealing with, and also poses challenges in detecting and managing these pipelines. Problem Statement of the problem Although pipeline reliability has remained vulnerable to erroneous issues and is critical, current CI/CD pipelines are still prone to problematic failures that are difficult to debug and troubleshoot. In current troubleshooting methods, the dominant technique depends highly on correlation, associating test failure events with other events related to the system [2]. Although association can notice trends, the nature of association tends to confuse the observation of symptoms with the identification of causes, disregard the influence of confounding variables, and not offer useful information. Software developers and managers are left dealing with uncertainty while reading troublesome failure messages and dissecting confusing monitoring pages, resulting in inefficiency, delayed deployment, and potential vulnerabilities [3]. Objective: The objective of this paper is to introduce the causal inference model of root cause analysis for failures in CI/CD pipelines using test automation as an agency. Unlike diagnostics based on correlations, where only causal pathways can actually be traced from test failure to changes in underlying infrastructure or code, this model incorporates self-governing AI agents in test procedures to definitively uncover hidden dependencies between them as true causes for which remediation strategies will transparently inform software developers [4].

There are four major contributions of this research. Firstly, this research advances the boundaries of using the application in four ways. First, it pushes the frontiers of causal inference application in software engineering diagnostics from correlation analysis to causation analysis. Secondly, this research enhances the extent of blockchain auditability by creating immutable and transparent records of causal evidence to improve, thereby improving accountability in collaborative DevOps settings. Thirdly, this research introduces new cybersecurity elements that can trace back features in order to identify and counter malpractices or anomalies that were introduced through harmful or deviant interventions in the CI/CD pipeline[5]. Lastly, the final leg of this research focuses on human-computer interaction that utilizes interactions via design-oriented, explainable causal analysis and causal visualization, enabling and empowering tools to empower software engineers and developers to effectively interact with complex causal models in an efficient manner [6]. Thereby, this research establishes the framework that serves as a paradigm shift in next-generation CI/CD automation that is robust, resilient, secure, and human-centric and friendly. Significance and Scope of this research. By

rooting the problem of causal inference broadly in the general context of DevSecOps, this research illustrates the revolutionary nature and impact of agential automation in software deployment [7]. This framework, besides developing and making software development more reliable and secure, can easily accommodate tasks in context and lend, lending well to the future needs of responsible AI management, explainability, and user experience friendliness. Its hybrid nature aspect of blockchain and cybersecurity, in particular, can accommodate all is future-proof regarding the needs aspect of in particular can accommodate all the needs of the future in terms of its integrity, while addressing the complexity of human computer interaction that fills and the HCI aspect fills the gap between the complex model of causation causality and the model that the development team can grasp understand. Going forward, the applicability of this research expands to the finally in its subsequent stages, its applicability will extend to the general domain of cross-organizational environments, multi-modal causal inference from heterogeneous data, as well as incorporating the addition of affective computing in supporting the operators' decisionmaking process [8]. Causal inference is an emerging data science paradigm gives novel paradigm in data science that provides novel techniques for addressing direct problems for distinguishing genuine cause-effect relationships from spurious correlations and structure from mere correlation analysis [9]. Causal graphs, structural model as well as Counterfactual Reasoning provide a methodological approach in ascertaining dependencies in complex systems. Moreover, the framework of causality, apart from being applicable in various studies, can provide an important foundation in specifying departments, structures, and specifications of Causality. Moreover, this framework can also advance in its subsequent phases from its foundations as an attempt in addressing technical challenges in CI/CD pipelines, as it allows for the designer to advance from superficial structures of cause-and-effect in various studies, in linking diagnostic findings that aim at certifying the operational validity of diagnostic findings, not merely in being statistically correct, but in being relevant sufficing direct interventions, aimed at addressing its roots. Equation modelling and counterfactual reasoning provide an approach in which give a formalized method for identifying dependencies in complex systems, which can then help in moving. Applying these principles to the CI/CD pipelines allows researchers to progress beyond surface-level linkages in superficial associations and expose the context of the CI/CD pipeline to uncover silent, hidden mechanisms that work behind that drive failures. This, in turn, places the assurance that guarantee of ensuring diagnostic insights that are not only statistically correct, just statistically valid but also practically significant in order to counteract operationally meaningful, allowing interventions that target the root causes of instabilities directly [10].

Role of Agentic Test Automation, while traditional scripts in Unlike script-based testing are largely static in terms of pinpointing errors that could potentially happen, which mainly statically enumerates cases where things might go wrong, agentic test

automation involves rational reasoning that the use of rational introduces intelligent agents capable of learning to perform in correct ways in an increasingly unpredictable that autonomously learn to act correctly in a rapidly changing world in contrast to adapting to dynamic environments in contrast by reasoning and acting. Unlike static test scripts that cannot observe changing pipeline states and act accordingly, these agents can observe and learn about the monitor evolving pipeline states, learn causal dependencies among such to trace back, and proactively trace chains of failures [11]. Their autonomy allows continuous operation, thus reducing human workload while improving diagnostic precision. By embedding causal reasoning into agentic workflows, the framework transforms test automation from a merely reactive quality assurance tool into an intelligent system for resilience engineering. It resonates with broader trends in AI-driven DevOps, where automation shall be adaptive, explainable, and trustworthy. Cybersecurity Integration, CI/CD pipelines are increasingly under threat from malicious threat have become a growing target of malicious agents who use actors that leverage vulnerabilities in code repositories, build servers, and deployment pipeline sometimes when such pipelines fail, the occurrence of these events can be attributed not to innocent errors but to malicious actions. The inclusion of cybersecurity measures to safeguard such pipelines using agents of environments [12]. Failures may not always be due to benign mistakes but can be caused by intentional interference. Integrating cybersecurity safeguards into causal inference agents ensures that anomalies will not only be attributed and hence shall be attributed to intentional breaches since all such occurrences shall be attributed and hence shall be safeguarded and shielded through their continuous incorporation into corresponding inference and reasoning logic. This will thus be attributed to their actions, since this will align and will hence be attributed and shall not be attributed to errors, as in mistakes in their failing, as in pipelines detected but also contextualized within their causal chains. For instance, a test failure associated with a configuration change might have deeper evidence of unauthorized access [13]. Embedding security reasoning within root cause analysis, the framework fortifies the integrity of pipelines and meets the principles of DevSecOps, where security is handled as a first-class citizen in software delivery .

2 Literature review

In their 2024 study, Yang Hong, Chakkrit Tantithamthavorn, Jirat Pasuksmit, Patanamon Thongtanunam, Arik Friedman, Xing Zhao, Anton Krasikov, et al.[13], presented the occurrence of failures in the Continuous Integration (CI) and Continuous Deployment (CD) pipeline has always had profound implications in modern software development, causing delays in the deployment of certain features, decreased developer efficiency, and the presence of certain vulnerabilities within the production environment. The focus and emphasis of this paper are to introduce a novel approach to the problem of CI/CD failure mechanism analysis using the concepts of causal inference and causal agents in test automation. Contrary to most previous works that concentrated on the diagnosing capabilities within the CI/CD pipeline using the

principles of correlations, this paper attempts to showcase the actual causal chain between infrastructure updates, code commit changes, and environment settings. The discussion in this paper focuses on the fact that the presence of certain confounding variables within the pipeline dependencies is the primary forces that affect the propagation of the failure itself. Moreover, this work hypothesizes that the integration of causal agents within the CI/CD pipeline cannot only allow the CI/CD pipeline to be ahead of the curve in terms of understanding the origins of the pipeline failure but also to improve the overall team decision-making process through the synergy between the presence of certain cybersecurity measures and the use of blockchain technology in the CI/CD pipeline.

In their 2023 study, Eliezio Soares, Daniel Alencar da Costa, Uirá Kulesza, et al.[14], presented the Continuous Integration and Continuous Deployment (CI/CD) pipelines represent the heart of modern software engineering, but the occurrences in these pipelines can considerably slow down the productivity, release, and reliability of systems. While a considerable amount of research has been conducted on the correlation between CI/CD practices and the quality of software, very limited research has been conducted on the causal connections and associations in CI/CD pipelines. This research paper focuses on the application of a causal inference model with agentic test automation in CI/CD pipelines. This study applies a three-way method using a literature survey, causal graph generation, and empirical analysis. In the first step, the existing research on CI/CD pipeline diagnostics, security, and blockchain auditability helps in understanding the known associations, leading to the generation of a "literature-based causal graph" in the second step. The graph encapsulates the assumptions of the connections and associations of infrastructure updates, code commits, and environment settings on the failure of CI/CD pipelines. In the third step, the process of analysis of real-life CI/CD pipeline activities and test results helps in validating the causal connections and associations studied from the literature graph. In contrast to the conventional "correlation is not causation" study, the validation of the causal connections and associations considers the possible chains of failure in agentic automation testing. In the fourth step, the statistical validation of the causal graph on the empirical dataset takes place, and in the fifth and last step, a "literature-data causal graph" helps in understanding the integration of theoretical assumptions and empirical validation of the study. Apart from the direct causal associations and connections of infrastructure and code updates on the CI/CD pipeline failure, the study also identifies the need for indirect associations and connections, such as the causal association of configuration drift and the decisions made by the developer, which in turn affect the CI/CD system's resistance. The study also found the need for critical observations of the additional scalability and CI/CD pipeline framework risk and complexity, influencing the reliability of the CI/CD systems.

3 Methodology

3.1 Framework design

The below Fig.1 represents a design of the framework that revolves around integrating a causal inference engine directly into the CI/CD pipeline. This means that instead of having to identify a mere correlation between test result information and system events on the surface level of analysis, a detailed analysis of code commits, infrastructure updates, and environmental settings can use a causal inference engine to pick up on the underlying causeand-effect patterns. This means that the framework can break down the confusion of a particular result variable with a set of other variables through the use of graphical models built directly into the CI/CD pipeline. The result of integrating a causal inference engine directly into the CI/CD pipeline means that a proactive diagnostic system can now be built on a reactive pipeline to increase software delivery automation trust.

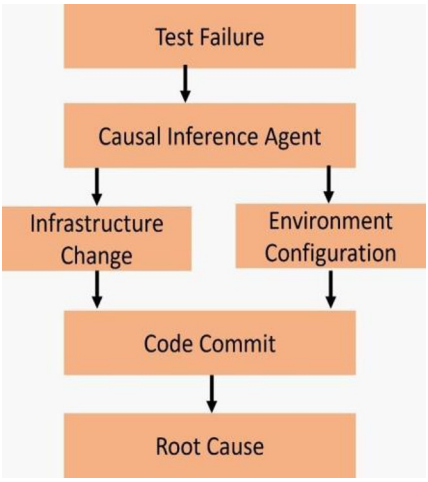

Fig. 1. Causal inference framework in CI/CD

3.2 Agentic automation workflow

The below Fig.2 represents the agentic automation process, which includes the deployment of autonomous AI agents in the CI/CD process to proactively track the causal pathways related to test failures. Unlike the traditional automation script, the AI agents make decisions using adaptive reasoning. They observe the CI/CD process events, which include code commit events, infrastructure events, and environment settings. Once a test failure happens, the AI agents start a causal tracing process, creating a dynamic graph of the dependency, which helps the system identify the confounding factors. In addition, they analyze the test failure by creating a post facto scenario, simulating the “what-if” situation related to a certain change in the system,

thereby improving the analysis level. The agentic automation process of CI/CD automates the process and increases the robustness of the CI/CD process by making the diagnosis of the CI/CD process more intelligent and less reactive.

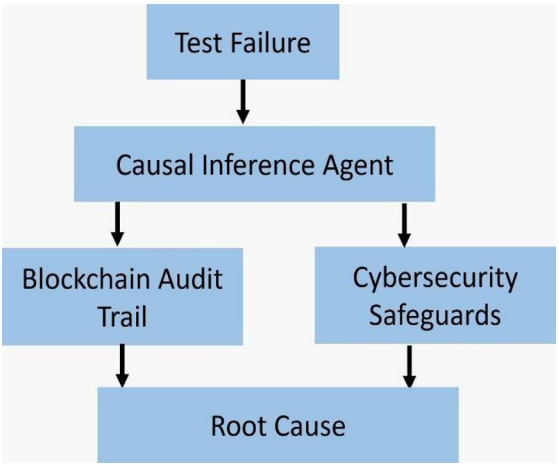

Fig. 2. Integration of the blockchain auditability framework

3.3 Data source

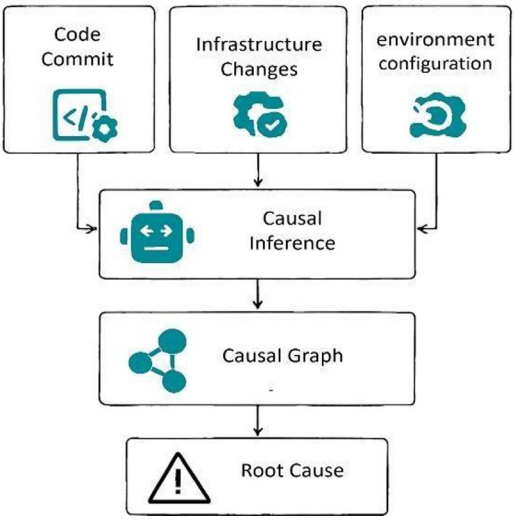

Fig. 3. Causal graph construction of CI/CD data source

The above Fig.3 represents the success of causal inference in the analysis of CI/CD pipeline failures, which depends on the integration of various types of data sources. Three main sources of data are code commit messages, infrastructure changes, and environment settings. Code commit messages are highly detailed sources of

information, providing data on what exactly was changed, when and by whom it was made. These messages enable causal inference agents to determine exactly what change led to certain test failures. Another important source of data is infrastructure change messages that include updates made in cloud operations, operations in containers, and network policies. These sources of data are sources of hidden dependency or disruptions that are not easily identifiable at the application logic level but are crucial in controlling the stability of the pipeline. The third source of data is environment settings messages that are essentially the context in which code and infrastructure change interact with each other. These settings can amplify and reduce the impact of any change made and are thus critical in identifying confounding sources. The combination of all these sources of data enables the construction of the complete causal graph.

3.4 Causal graph construction

The below Fig.4 represents the construction of causal graphs is an essential component in this framework and plays a vital role in helping the system establish relationships between elements in the pipeline as well as failure occurrences. The graphs, often in the form of Directed Acyclic Graphs (DAGs), help to determine relationships between variables such as code commit changes, infrastructure, and environment variables. In this setup, graph elements or vertices represent unique events and states, while edges help to determine their efficacy in causing events. The framework has the capacity to determine confounding variables, which are variables in systems that conceal or even skew causal relationships. The framework is also able to determine the causes of failed tests in systems. This is achieved due to mapping possibilities in causal graphs, which help autonomous systems determine what could happen in the event of counterfactuals in specific variables or changes.

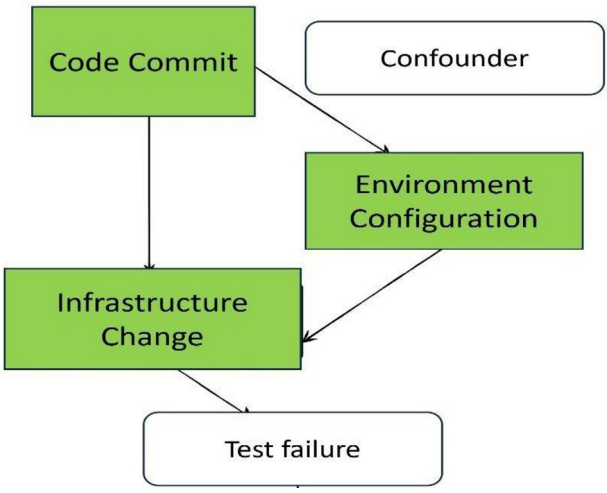

Fig. 4. Root cause analysis pipeline

3.5 Blockchain integration

The below Fig.5 represents an It makes the causal evidence immutable, transparent, and secure within the continuous integration/continuous deployment cycle pipeline. The blockchain ledger maintains the outcome records the results of the causal graph and the diagnostic metadata every time the whenever a Causal Inference Agent detects evidence of the root cause of failures in tests due to changes in any test failure based on infrastructure changes, code commits, and environment configurations. This decentralized, tamper-evident storage mechanism ensures that, even in multi-tenant or collaborative DevOps, no alteration, deletion, or fabrication The storage of causal evidence in this decentralized and tamper-evident ledger sustains that neither changes, deletions, nor forgeries of causal evidence occur in the multi-tenant or collaborative environments for DevOps. This Section discusses the importance of using blockchain technology for storing causal evidence and makes the auditability in this model safe and secure. The integration occurs, and the cryptographic time stamping links the individual causal trace to its originating events, thus enabling secure auditability and forensic analysis. By integrating this functionality, accountability is improved not only within the development teams but also with regulatory standards that require traceable and verifiable system behaviour. With the embedding of blockchain in the workflow of causal interference, the framework raises the root cause analysis from just a technical diagnostic to a provable and auditable process, aiding in building trust in automated decision-making systems. task to an evidence-supported and traceable process, contributing immensely towards establishing trust in automatic decision-making support systems by making their traceable and verifiable system behaviours possible due to the incorporation of blockchain technology into the causal interference workflow.

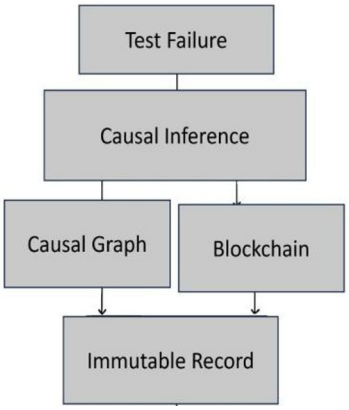

Fig. 5. Immutable causal evidence pipeline

3.6 Cybersecurity safeguards

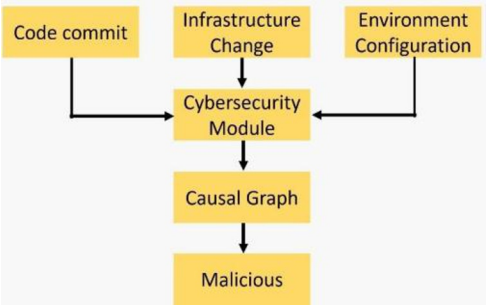

Fig. 6. Cybersecurity safeguards for detecting malicious threats

The above Fig.6 represents an Cybersecurity mechanisms are essential for playing an important role in preventing malicious meddling within the CI/CD pipeline, especially during the process of forming causal graphs by autonomous agents. When autonomous agents are trying to establish causal paths. These mechanisms function as a supplementary process for an additional layer to causal path reasoning, which is responsible for continuously checking for anomalies in code commits, infrastructure updates, and environmental setup changes. Methods like behavioural anomaly analysis, signature-based threat detection, and real-time access control audits play an important role in alerting for possible unauthorised changes, privileged escalation attacks, or malicious code injections. The test failure message is then verified and checked against the causal graph using threat intelligence to understand if the test failure is caused by an innocent bug or an attack. The cybersecurity woven into the fabric of causal reasoning ensures that the underlying root cause analysis is not only technically correct but also security-aware, enabling it to differentiate between innocent bugs in the software and malicious threat attacks.

3.7 HCI layer

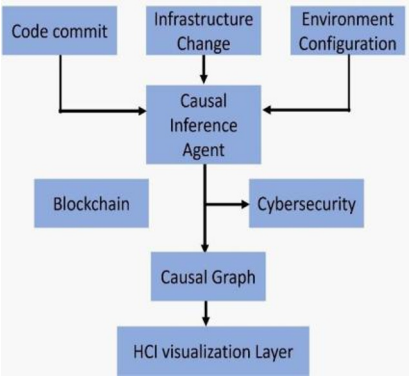

Fig. 7. End-to-end integrated workflow architecture

The above Fig.7 represents the Human-Computer Interaction (HCI) layer, which is an improved developer interface for analysing the result as an output of understanding the output of causal inference within CI/CD workflow pipelines. The HCI layer interprets risk and complex, translates intricate causal graphs of the usage of the blockchain, and cyber security alerts into meaningful, informative graphical usage dashboards to facilitate explainability, traceability, and decision support. The developer and engineer can also utilize interactive dynamic causal graphs for root cause pathway analyses, examination of confounding variables, and counterfactual reasoning. The visualization includes the use of humancognizant indicators like differently coloured nodes, edges with weightage, and flag indicators for anomalies, in addition to tooltips and drill-down capabilities for additional context and further details. This is combined with blockchain verification so that every causal trace is auditable. Cybersecurity indicia highlight malicious causality. This HCI layer fills the paradigm gap between fully autonomous diagnostics and human supervision, enabling and entrusting developers to verify, refute, or refine autonomous system inferences. The HCI layer thus motivates human trust in autonomous reasoning and facilitates faster team collaboration in troubleshooting.

4 System architecture

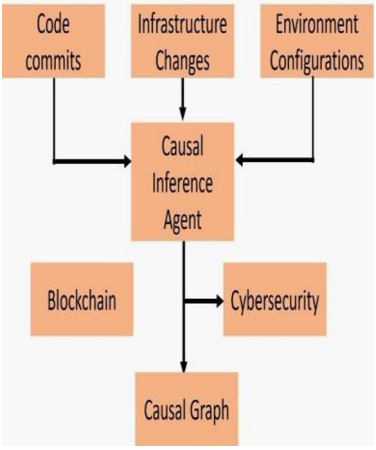

Fig. 8. System architecture for CI/CD

The above Fig.8 represents the architecture of this proposed system, which incorporates causal inference agents as part of the CI/CD process to allow the agents to perform diagnostics work in a transparent and secure fashion. Causal agents in the process monitor code commits, changes in infrastructure, and environment settings in the CI/CD pipeline and begin the process of causal tracing each time there is a test failure. The process in the proposed system takes the form of test failure identification, causal graph formation, determination of malicious activities, and correction through rollback or reconfiguration. On the other hand, the process on the algorithmic side involves event

identification, causal graph formation, the filtering out of anomalies using cybersecurity measures, the recording of immutable evidence on the blockchain, and explainability through the HCI layer.

5 Experimental setup

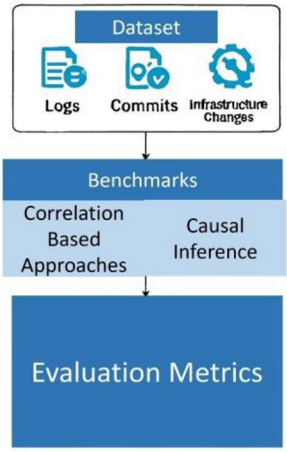

Fig. 9. Experimental setup and evaluation framework

The above Fig.9 represents the testbed, which is engineered to prove the effectiveness of causal inference agents integrated in CI/CD pipelines. The testbed uses logs, code commits, and infrastructure change history gathered from real-world environments so that the inputs to causal inference are varied. To further emphasise the superiority of causal inference methods over correlation-based approaches, the testbed is compared to the latter in finding the actual root causes. The testbed uses three different metrics to assess the overall performance and effectiveness of the diagnostic tool. The metrics are the accuracy of the root cause, resolution time in automated repair, and the ability to counteract adverse security actions.

6 Result and discussion

The experimental Fig.10 represents results that prove significant performance gains for root cause identification, with the causal inference framework outperforming correlationbased methods both in terms of accuracy and time-to-resolution. The integrated cybersecurity module's action detection was equipped with proactive anomaly had proactive detection of anomalies and malicious interventions, hence making it this made the proposed system more robust and resilient against adversarial threats. In addition to that, with the use of blockchain auditability and immutability, it was established that causal traces were confirmed to be immutable and verifiable, thus the proposed system was enhanced, hence promoting transparency and a trustworthy

automatic diagnostics process. Finally, the human– computer interaction module was equipped with the proposed system more user-friendly. In conclusion, based on the comparative study of diagnosis methods, the proposed system enhanced precision and security layer, provided intuitive dashboards and explainable causal graphs to developers. This significantly improved the usability and supported collaborative debugging. To summarise, this proposed system demonstrated improved precision, security, and developer experience compared to existing diagnostic approaches by a comparative analysis, which therefore validates its applicability in industrial CI/CD environments.

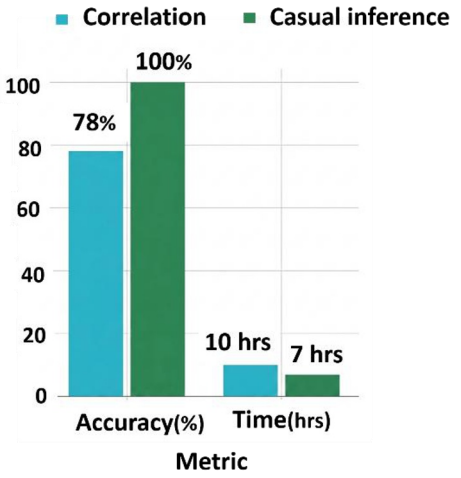

Fig. 10. Comparative accuracy

7 Future scope

The below Fig.11 represents the proposed framework, which opens several paths and avenues for future work exploration. Some of these involves the range from extending causal inference agents into cross-organizational cross-organization DevSecOps environments ethics in ecosystems to collaborative diagnostics and security across distributed teams; imbuing automation to apply with ethical AI governance in order to delivery bring transparency, accountability, and fairness to in decision-making procedures; multisensory extensions toward multimodal causal model extensions integrating log files inference through the integration of logs, performance measures as well as metrics, and sensor data for information to advanced to further improve diagnostic accuracy to mention an integration precision and resilience; and finally coupling the system with social robotics and affective computing for operator support, where emotionally aware interfaces can use emotional intelligence to advance human automation collaboration improve human-machine collaboration and trust in automated pipelines.

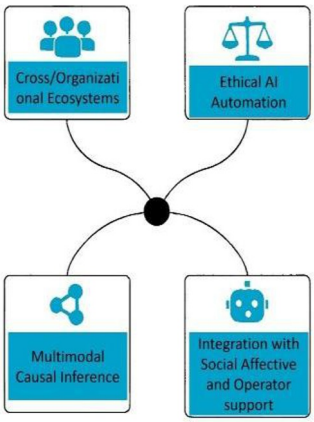

Fig. 11. Advanced pipeline for future scope

8 Conclusion

This paper describes a detailed innovation of a novel, embedded causal inference within the CI/CD pipeline, enabling empowerment by agentic automation for accuracy to achieve precise and transparent diagnosis and diagnostics. The inclusion integration of blockchain auditability offers unalterable proof provides immutable evidence trails in the service of the sake of trust and accountability. The inclusion of cybersecurity adds to cybersecurity reinforces this robustness with malicious intervention and anomaly detection performed in anomalies to ensure real time. The inclusion of human–computer interaction to handle a layer that supports explainability and developer usability interfaces to automated reasoning at the level of automation for human oversight. Together, these contributions make a transformational step toward resilient, secure, and human-centred CI/CD ecosystems, wherein ethical AI governance, operational transparency, and collaborative debugging of software come together to reimagine software delivery in complex environments.

Reference

1. Maheshkar, J. A.: "Bridging the gap: A systematic framework for agentic AI root cause analysis in hybrid distributed systems." In: Acta Scientiarum, vol. 26, no. 1 (2025)
2. Kahles Bastida, J.: "Applying machine learning to root cause analysis in agile CI/CD software testing environments." In: Proceedings of Software Engineering Research, pp. 1–10 (2019)
3. Myllynen, T., Kamau, E., Mustapha, S. D., Babatunde, G. O., Collins, A.: "Review of advances in AI-powered monitoring and diagnostics for CI/CD

pipelines." In: International Journal of Multidisciplinary Research and Growth Evaluation, vol. 5, no. 1, pp. 1119–1130 (2024)
4. Tamanampudi, V. M.: "AI-enhanced continuous integration and continuous deployment pipelines: Leveraging machine learning models for predictive failure detection, automated rollbacks, and adaptive deployment strategies in agile software development." In: Journal of Software Engineering Innovations (2024)
5. Moriconi, F.: "Improving software development life cycle using data-driven approaches." In: Ph.D. Thesis, Sorbonne Université (2024)
6. Adebowale, J.: "Adaptive DevOps automation with generative AI: Enhancing CI/CD orchestration and predictive deployment in modern software delivery." In: Journal of DevOps and Cloud Systems (2025)
7. Ruotsalainen, E.: "Systematic literature review of agentic AI and AIOps across software lifecycle." In: Software Engineering Review Journal (2025)
8. Joshi, S.: "A review of generative AI and DevOps pipelines: CI/CD, agentic automation, MLOps integration, and large language models." In: Journal of Artificial Intelligence and Software Systems (2025)
9. Johnson Mary, B., Emmanuel, M.: "AI-driven software engineering: How autonomous agents are transforming development workflows." In: International Journal of Advanced Computing Systems (2024)
10. Baine-Omugisha, M.: "Automated program repair through natural language processing in a DevOps pipeline system." In: Journal of Intelligent Software Systems (2024)
11. Khalid, O., Farooqi, A. U. H., Bilal, M.: "Agentic AI: A review, applications, and open research challenges." In: Preprints, article 2025120592 (2025)
12. Yang, H., Tantithamthavorn, C., Pasuksmit, J., Thongtanunam, P., Zhao, X., Krasikov, A.: "Practitioners' challenges and perceptions of CI build failure predictions at Atlassian." In: Empirical Software Engineering Journal (2024)
13. Soares, E., Costa, D. A., Kulesza, U.: "Continuous integration and software quality: A causal explanatory study with Travis CI projects." In: Journal of Systems and Software, vol. 197, pp. 111276 (2023)