International Journal of Advanced Engineering Science and Information Technology (IJAESIT)

|ISSN 2349-3216| Volume 9, Issue 2, March – April 2026| Bimonthly, Peer Reviewed and Scholarly Indexed Journal|

DOI: 10.15662/IJAESIT.2026.0902004

AI-Assisted Adaptive Circuit Breaker Monitoring and Failure Prediction for Distributed Cloud Platforms

Dr Somasundaram Krishnan

Professor, Department of Computer Science and Engineering, Sri Muthukumaran Institute of Technology,

Chennai, India

Publication History: Received: 11.03.2026; Revised: 12.04.2026; Accepted: 17.04. 2026; Published: 20.04.2026.

ABSTRACT: Fast and intelligent fault detection and recovery for cloud systems (distributed and composed of dynamically changing microservices, virtualisation and/or geographically distributed nodes) is a big concern. The next research paper presents a framework for adaptive monitoring and prediction of the failure of circuit breakers in order to increase the resilience of distributed cloud environment. It is not just a framework for collecting real time data for telemetry, it also is there to keep an eye on services, identify anomalies and model failure prediction and adaptively control ‘circuit breakers'. Using machine learning methods these operational metrics such as latency, error-rate, request throughput, CPU/ram load, network delay and dependency behavior of the called services are continuously captured and analysed. The predictive actions layer is able to identify failure trends before they can lead to service outages, while the adaptive decision layer can dynamically adapt the circuit breaker levels to take into account the variability of traffic, past failures or workload trends. This unique architecture will allow for proactively forecasting and automating fault isolation to minimize subsequent cascading fault events and thereby providing a high level of service availability and self-healing to a cloud service. In summary, this work demonstrates the AI potential to disrupt the conventional fixed circuit breaker paradigm, bringing intelligence, context and reliability to this solution. It provides a scalable and efficient way of making a system more fault tolerant, reducing downtime and ensuring reliable performance in today's distributed systems, typically deployed in the cloud.

KEYWORDS: Artificial Intelligence, Adaptive Circuit Breaker, Failure Prediction, Distributed Cloud, Fault Tolerance, Anomaly Detection, Self-Healing Systems.

I. INTRODUCTION

Scalable computing, flexible resource allocation, real-time data processing and applications delivery, money savings – wherever, and an essential to the current digital services – are just a few facets of distributed cloud platforms. Distributed cloud is an alternative to the traditional centralized cloud concept, and consists of a distributed set of interdependent data centers, edge nodes, microservices, containers, virtual machines, storage systems and network elements that function as a single end user experience cloud. They're essential to the stability and reliability of e-commerce systems, banks, health-care systems, industrial automation, smart cities, online education, streaming solutions and enterprise applications where a high availability system is required, and a quick and accurate response is needed and expected. With more platforms being distributed in the cloud and the more complex service architectures, however, the more things can go wrong – and thus the more that the service can degrade and cascade failures can happen. Even the simplest case of failure of one microservice, network link or resource node may impact other microservices that the microservice itself reliant on, which may in turn result in a major disaster if they are not timely detected and managed.

One of the most serious issues with distributed cloud platforms when confronted with uncertain and dynamic conditions is reliability of the system. Cloud workload is transitory, with the number of users that need to request resources and use them, and the type of traffic or services they exchange changing constantly and evolving. During peak time periods, the cloud system may be suffering from additional issues such as latency, request timeout, high error rate or memory overload/CPU saturation with network congestion. Most fault handling approaches have been based on a threshold level and most of the proactive approaches are not suitable in such a case. The approaches used are essentially time-tested and can only find failures after-the-fact and can also lead to delays in the remediation, poor user experience and loss of service. Hence, a new paradigm is needed to customize adaptive and predictive failure management techniques intelligently to predict the early symptom of the faulty in modern cloud platform before reaching to critical level.

|ISSN 2349-3216| Volume 9, Issue 2, March – April 2026| Bimonthly, Peer Reviewed and Scholarly Indexed Journal|

DOI: 10.15662/IJAESIT.2026.0902004

Distributed systems can use the circuit breaker pattern as a very useful resilience pattern not to make repeated invocations to a failed service. A software circuit breaker will block any call that might be sent to the service that is failing or doesn't respond in a timely manner. Its normal functioning is to be closed, open or half open. If it's in "closed" then a request can commence again. When the failure rate is greater than a per cent or delay to reach system is greater than a per cent, breach changes to "open" and other requests fail to reach the system, thus saving the system from overload. After that, it is put into half-open condition which enables fumbling visitors to be sent for probing, if the service is back up. If a good service is encountered the breaker is closed, if not, it does not close. The virtue of this mechanism is that it can be used, the conventional approach to using it is rather static and rule driven. The service behaviour and failure characteristics of a dynamic cloud can often be unsuitable for fixed thresholds, when the workload intensity in dynamically changing.

One such possibility if the crash mechanism were to be made more effective would be by leveraging Artificial Intelligence (AI) of course, specific to a distributed cloud platform. Real-time operational data can be processed by AI-based models, detected for patterns which are less commonly seen, predictions can be made on the likelihood of service to malfunction and decision-making capabilities can be automated. A machine learning algorithm will be able to detect patterns and relationships by crawling through the performance data, expected queries, latency data, error logs and resource utilisations data – which is just too much to manually inspect. However, it is not impossible to achieve, as the shift from reactive (failure handling mode) to proactive (prevent failure mode) can be made utilizing a cloud based platform and by using an AI-assisted prediction. The system can detect whether or not there are potential problems with the service: latency is about to reach threshold; number of timeouts is increasing; CPU is unusually high; memory leaks; that request is failing; or that request is causing an error failure. This may enable the circuit breakers to act in a more intelligent manner, thus avoiding failures of dependent services normally caused by whipsaw.

In this research paper titled as "AI-Assisted Adaptive Circuit Breaker Monitoring and Failure Prediction for Distributed Cloud Platforms", the authors attempt to improve an intelligent framework to incorporate the adaptive circuit breaker control and failure prediction leveraging AI. The proposed design is to continuously monitor the health of various services, leverage the telemetry data collected, report anomalies, predict failures, adaptively tune circuit breakers based on the cloud environment and so on. It consists of collection layer for collection of telemetry data, Monitoring and pre-processing Module, AI based Anomaly detection model, failure prediction model, Adoptive decision layer and Automated circuit breaker control mechanism. These all factor into real-time reliability management, trying to keep different distributed cloud systems available when parts of the services are unavailable, and unpredictable workloads exist.

The relevance of this work is that it provides a solution to three key issues arising in the context of distributed cloud reliability management, namely the delayed detection of failures, static configuration of circuit breakers and cascading service disruption. Rarely service failure happens in isolation of other service failures in large scale cloud environments. A slow response time is caused by slow/depleted service causing many queued requests, leading to increased load across service assets, dependent services or ones on which the service depends and ripple effects for the entire application. An adaptive AI-assisted circuit breaker can minimize this risk by throwing those unhealthy services at the right time, definitely saving the healthy ones and, of course, providing those chances of healthy service without impacting the system. This increases the fault tolerance, reduces downtime and results in better user experience.

Moreover, using AI in circuit breaker monitoring fosters self-healing operation, which is a key strategy to decrease unplanned downtime in cloud systems. With Self Healing System, a problem can be identified, an appropriate corrective action can be taken and it can be automatically recovered back to a state of normal operation with less human effort. This capability proves to be very convenient in distributed cloud scenarios, with lots of services, nodes and entanglements, making manual monitoring difficult. AI can be used for learning and adapting the system to new pattern of failure and workload changes with the help of use of tracking to adapt decisions with AI.

|ISSN 2349-3216| Volume 9, Issue 2, March – April 2026| Bimonthly, Peer Reviewed and Scholarly Indexed Journal|
DOI: 10.15662/IJAESIT.2026.0902004

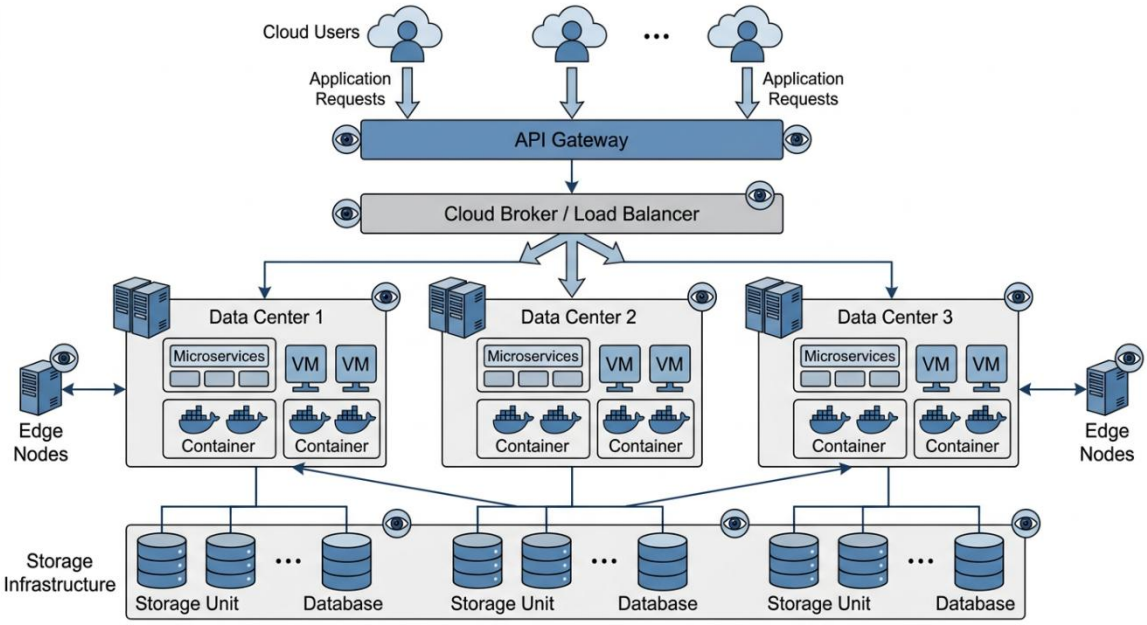

Figure 1: Architecture of Distributed Cloud Platform

This study overall provides to the area of cloud reliability engineering, a proactive and adaptive way of failure prediction and circuit breaker management. The paper highlights that this effort has the potential to transform the traditional circuit breaker paradigm to become a smart resilience element using AI. Together with real-time monitoring and predictive analytics, adaptable threshold adjustment will be part of the proposed framework that can increase availability for distributed cloud platforms, minimize service interruption and boost services' dependability.

II. RELATED WORK

Reliability issues of the microservice world have become a primary research topic including topics such as fault analysis, anomaly detection, fault diagnosis and automatic diagnosis of performance failures. Based on an industrial survey, benchmark system and empirical study, Zhou et al. studied fault analysis and debugging in microservice system and presented that the failures of a microservice system are hard to diagnose as it can be caused by service dependency, configuration mistakes, resource bottlenecks, network delay, or unexpected interactions among distributed parts [1]. The applicability of their work can be found in the fact that their work reaffirms a problem that is relevant to this research area – the fault propagation in the microservice-based cloud platforms – and the need of automatic fault monitoring and diagnosis.

But, in case of home electronics sector, Ganesan explored other fields considering the home electronics industry requires some smart service, automation and some adaptive decision making in the service environment by adopting artificial intelligence (AI) in customer self-installation experience [2]. For it is not a particularly deep study into the overall concept of the cloud circuit breaker, however, it is a good example that shows the possibility of using AI to improve the UX, manual tasks and even the operational efficiency. Zhao et al. also came up with a multimodal approach for anomaly detection to detect bad soft changes in the on-line service system [3]. They are pertinent because in today's distributed cloud platform applications are always being updated and if a change isn't correct, applications could fail or slow down.

Hierarchical causality graphs [4] were used to get their automated end-to-end performance diagnosis approach for cloud systems — called CauseInfer. As in this study, the similar method, many like the proposed framework, causal diagnosis will be used to reveal why service degraded and that it was on the verge of breakdown. Jin et al. suggested a robust principal component analysis based anomaly detection algorithm of microservice architecture [5]. In their research, they have shown that anomaly detection – one of the main considerations of the layer of the proposed framework to deal with anomaly detection – can be effectively provided through the use of statistical and machine learning.

Based on the above, Wang et al. suggested that the workflow can be utilized to build a statistical method based automatic fault diagnosis approach for microservice based applications [6]. Their research is relevant in that focusing

|ISSN 2349-3216| Volume 9, Issue 2, March – April 2026| Bimonthly, Peer Reviewed and Scholarly Indexed Journal|

DOI: 10.15662/IJAESIT.2026.0902004

on the awareness of existing workflows aids the understanding of failures across the service chains and how problematic services are dependent on other services (this is extremely important when trying to prevent cascading failures). Raeiszadeh et al. presented distributed trace for real time anomaly detections of this microservice cloud apps [7]. They have directly used their findings with distributed traces, latency information and service level monitoring as failure predictors in real-time.

Gan et al. introduced the system of Seer that is used to perform debugging of cloud microservice performance via big data [8]. In this piece the authors discuss the relevance of operational, big information for multi-layered insights into the performance problems. As an instance of anomaly detection/classification, a typical one is using distributed tracing, such as Nedelkoski et al. [9] leveraging deep learning models to find hidden patterns in the failure data present in tracers to detect anomalies. In conjunction with this area, Bogatinovski et al. added the area of self-supervised diagnostics of anomalies from distributed traces [10] to be utilized, if data on labelled anomalies of failures is very sparse.

For microservice trace anomaly detection, Liu et al. proposed the involving service level deep Bayesian network that is an unsupervised method [11]. They are probability based and hence they can be used for failure prediction because uncertainties of the service behaviours can be estimated. To finish, Meng et al presented a means of comparing the execution traces to detect anomalies in microservices [12]. It is a complement to the concept of "getting trace level comparison" which entails looking and locating a different path to take that is not the one that was normally taken and could end up at an abnormal situation and therefore trigger the adaptive circuit breakers before the failure would end up at one of the other distributed cloud platforms.

III. AI-ASSISTED ADAPTIVE CIRCUIT BREAKER MONITORING AND FAILURE PREDICTION

AI- Assisted Adaptive Circuit Breaker Monitoring and Failure Prediction for Distributed Cloud Platforms is a project that aims to boost the reliability, fault tolerance and continuity of services in a complex cloud by leveraging adapted circuit breaker technology and artificial Intelligence. There are various kinds of microservices, virtual machines, containers, edge nodes, storage units, load balancers and network links that all are interconnected to form a distributed cloud platform. These still result in a lot of exchanges between them, on demand and on reply, and an ever increasing dependence between each individual service becomes apparent. These services can be heavily dependent on another and if one service fails, it might take down a whole bunch of other services. This results in an ideal approach – anomaly monitoring and warning in real time by AI algorithm, failure prediction analysis and even adaptive circuit breaker control, among others, so as to sound the alarm before a trouble becomes a services issue.

The scheme is an architectural body, which is multilayered, structured and organised. This consists of five components: data collection layer, monitoring and data pre-processing layer, anomaly detection and further prediction layer using AI, failure prediction layer and decision layer for adaptive circuit breaker. These layers work together to continuously observe cloud service behavior, detect abnormal behavior, predict possible failures and change the behavior of circuit breakers whenever necessary. The proposed work, however, which is suitable for intelligent learning models, adds a context aware decision making depending on the state of the actual system.

3.1 Data Collection Layer

There are two levels in the framework: Data Collection layer. This layer is responsible for gathering the data of the distributed cloud platform's operations across its various components. These can be data on the time it took to respond to a service, the number of timeouts against a service, the latency of its requests, latency across the service in general, data on through-put, error rate, patterns of failure requests, data on logs on service dependency and failed requests, CPU usage, memory usage, disk I/O, network delay, number of packets lost, and also the number of container restarts. It all comes to you in the forms of a cloud of monitoring agents, application performance monitoring applications, container orchestration solutions, a service mesh of logs and distributed tracing systems.

For distributed architectures some of the services may be fine when workloads are high, but failing when there is a lack of resources and/or when there is lots of traffic being requested, and/or dependency concerns. Therefore the framework is not based on a single parameter to determine the health of services. Rather it collects multi-dimensional telemetry information at multiple points in cloud infrastructure. For instance, a high latency could be due to congestion or any of the following in the network: new requests, delay or overload in the database. Multiple metrics need to be used in the framework to get a better understanding of the services behaviour and thus minimize false detection of failure.

|ISSN 2349-3216| Volume 9, Issue 2, March – April 2026| Bimonthly, Peer Reviewed and Scholarly Indexed Journal|

DOI: 10.15662/IJAESIT.2026.0902004

3.2 Monitoring and Preprocessing Layer

After the process of collecting information, it will be transferred to the monitoring and data pre-processing/aggregation process layer which will preprocess the information for its analysis by AI. Cloud monitoring information is typically multifaceted, noisy, incomplete and inconsistent. In this perspective, it is necessary to quality the data inputs which is performed by using pre-processing technique. The processing in this layer cleans out the data, normals having missing data, cleans noise out, and can extract feature segments and time windowing segments.

Time windows for which collected metrics is fixed to analyse behaviour over time. For example, for the same defined time periods, 30 seconds, 1 minute or 5 minutes could be defined, and the following metrics analysed - latency of application requests, error rate or CPU utilisation, depending on the requirement. One of the advantages of time window based monitoring is that once a failure state starts to happen, the framework can tell – even if it is a gradual failure over time – and not just when the failure happens suddenly. This is essential since a number of cloud failures happen over time. Rising latency is the first symptom of the issue followed by one or more timeouts, error counts and, finally, no service.

Furthermore, in this layer feature extraction is carried out. The important metrics like the Average Latency, the Peak latency, the percentage of errors, the ratio of getting a success, the resource saturation level, the moving average, how often it fails, the Dependency latency are computed from the raw metrics. These features are in turn fed to the AI layers detection/prediction.

3.3 AI-Based Anomaly Detection Layer

Anomaly detection Layer compares the behaviour of services being monitored against the learnt normal behaviour of services, detects the abnormal service behaviour. The usual way of detecting the anomalies is by the use of the pre-defined threshold values in the traditional monitoring systems. But in the distributed cloud space such normal operation can vary, based on the time of day, load on a platform, user's use of a platform or resources. If, at a particular time of day, you're really busy, there might not be any difference in the latency, or, at a time when you're not so busy there will be and that would be abnormal.

The proposed framework tries to overcome this limitation by the application of anomaly detection using AI. Machine learning models that have been trained with past and current metric measurements about the normal service behaviour. Based on the size and complexity of the cloud platform, algorithms can be applied such as Isolation forest, random forest, support vector machine, autoencoder or hybrid deep learning models can be used. The model detects deviations with regard to the behaviour of the resource use and service dependency of latency, throughput and error rates.

This layer will provide a piece to the resulting Anomaly Score. When the anomaly score is low, it means that the service's behavior is normal, and when the anomaly score is high it means that service behavior may degrade. The anomaly score is not only used to open circuit breaker instantly. Rather it is given to the failure prediction layer for further analysis. This prevents the circuit breakers from false tripping and unnecessary tripping of the breakers.

3.4 Failure Prediction Layer

The key knowledge component of the proposed framework is failure prediction layer. It's used to estimated the risk of potential service failures for the future. Failure prediction is identifying abnormal behaviour(s) that could cause a failure of service.

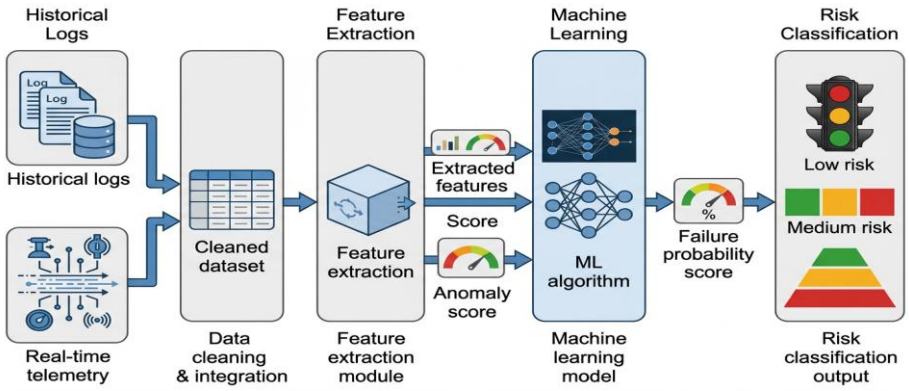

Figure 2: AI-Based Failure Prediction Workflow

|ISSN 2349-3216| Volume 9, Issue 2, March – April 2026| Bimonthly, Peer Reviewed and Scholarly Indexed Journal|
DOI: 10.15662/IJAESIT.2026.0902004

It utilizes the failures that have occurred in the past, telemetry data, anomaly scores and service dependency to predict future failure risk in this layer. It considers growing latency, overlapping timeouts, falling throughput, unexpected increase in the speed of growth of mem usage, percentage of CPU usage and network instability/container restarts. These patterns are then used to give to each of the monitored services a probability of failure (PoF) score.

Failure Prediction Model: If a microservice is constantly experiencing significant rise in its response time, significant rise in number of errors per time period, high CPU consumption and multiple calls from microservices to that microservice have failed during these runs, then Failure Prediction Model identifies it as High Risk Microservice. But if there are no other metrics, and just one is temporarily high it can be considered as less dangerous. It is a forward-looking approach to differentiate failure trends and transients.

The service dependency relationships are also taken into account by the failure prediction layer. Services in a distributed cloud platform can have multiple services on which they are dependent. Many dependent services may become unstable if a database service or authentication service starts to fail. So, dependency-aware prediction allows the framework to not only pick up direct risk of failure, but potential cascading effects.

3.5 Adaptive Circuit Breaker Decision Layer

The circuit breaker is in three modes of operation. Served normal number of service requests, in other words, the service is considered healthy. Fall-back mechanisms are employed to requests directed at the unhealthy service or requests are blocked temporarily. This way no more failed calls are attempted and other services are not impacted. Half open: Limited test requests allowed, to determine if the service has been recovered. If the test command is successful the circuit breaker is put in the closed state. If they don't then there will be extended recovery time available.

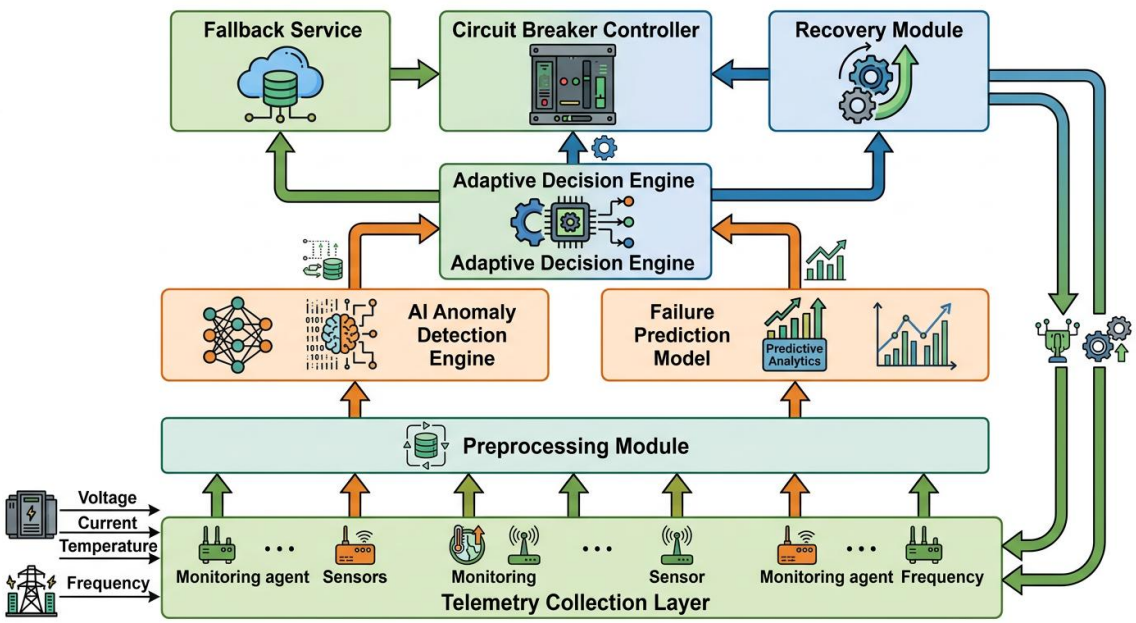

Figure 3: AI-Assisted Adaptive Circuit Breaker Framework

3.6 Feedback and Continuous Learning

Feedback is an important aspect in the proposed model. All the divercisive/action not actions are recorded and and input into the prediction accuracy as to when the action was taken by the circuit breaker. If the result from the model is correct and this is a prediction of failure, the prediction is considered to be a successful prediction. When the system checks-out the breaker for no reason, is considered a False Positive. If no prediction is made and a failure occurs, then the failure is considered as a missed prediction. In addition, Feedback Records are used to re-train and update the AI models.

This ongoing learning journey can help the framework evolve and add new workload patterns to it as new services behaviours are added, or as software or infrastructure changes, and find new configurations that still fail - adding them to the framework as well. This renders it suitable for dynamically changing clouds that can have quickly-aging policies.

|ISSN 2349-3216| Volume 9, Issue 2, March – April 2026| Bimonthly, Peer Reviewed and Scholarly Indexed Journal|
DOI: 10.15662/IJAESIT.2026.0902004

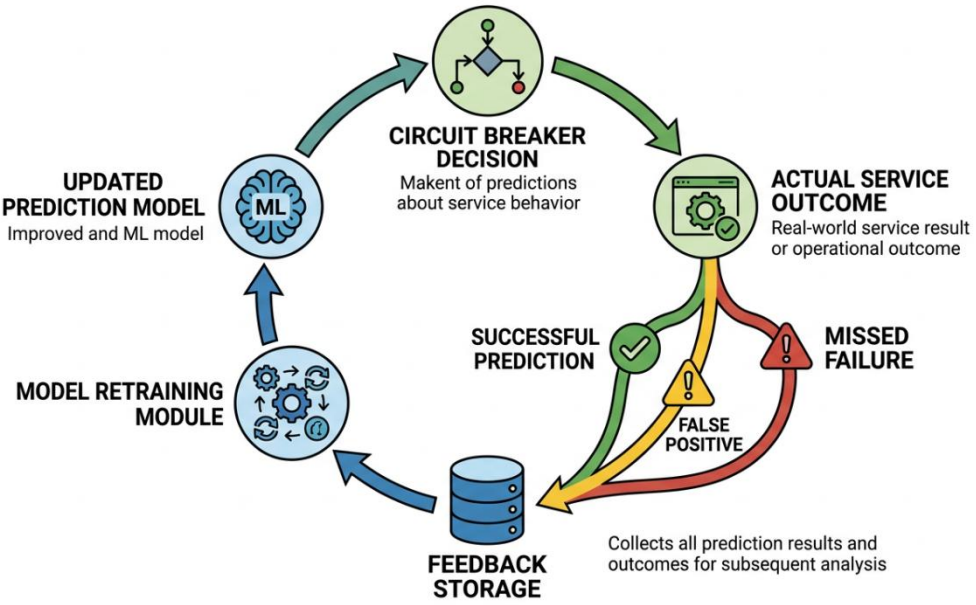

Figure 4: Feedback and Continuous Learning Cycle

3.7 Framework Outcome

The proposed framework 'scenario' is proactive, adaptive and intelligent towards reliability management for distributed cloud platform. This combines AI capabilities of monitoring, and adaptive controls of circuit breakers to ensure that failures are picked up before cascading failures occur, thereby providing more available services and enabling self healing cloud operations. It gives intelligence to the circuit breaker that not only detects when it's about to fail, but makes decisions about what it's going to do based on up-to-the-minute information on conditions in the cloud environment.

IV. FRAMEWORK EVALUATION

4.1 Evaluation Objective

The success of the innovative adaptive circuit breaker providing infrastructure (with the help of AI-driven FailSafe) is evaluated in terms of reliability improvement, failure prediction and continuity of services, in the context of a distributed cloud platform. The effectiveness evaluation of the framework to early discover abnormal behaviour of a service, predict a possible failure and trigger the circuit breaker to avoid a possible domino effect for other dependent services should be done. It is mainly to compare the effectiveness between the adaptive mechanism developed using the AI model and the traditional static circuit breaker solutions.

4.2 Evaluation Environment

It is compatible with (and may be used along with) other microservices, virtual machines, containers, edge nodes, databases and load balancers and thus can be used for testing the framework in a mock-up distributed cloud infrastructure. Each service is equipped with workload presets that simulate how the application request(s) in each service will be processed during normal service traffic, peak load, network delay, CPU overload, memory saturation and service timeout. Time-to-time information from agents (latency of requests, throughput, error, response time etc.,) along with CPU, memory utilization and the network latency and failure are monitored with monitoring agents having services as dependency. The measurements are sent to the anomaly detection and failure prediction models.

4.3 Evaluation Metrics

Prediction Based measures as well as System level measures are utilized in the assessment of this framework. To measure the accuracy of prediction the measures: accuracy, precision, recall, f1-score and false positive rate (FP rate) are used. These are useful indicators of the accuracy of the AI model when recognising services that are likely to fail. The Mean Time to Detect Failure, Mean Time to Recovery, Service Availability, Request Success Rate, Downtime Reduction and Cascading Failure Prevention Rate is how the system reliability is measured. The less the time that it takes for detection/recovery means the more quickly the framework will respond to loss of services. The lower the number the more the broker will be cloudy and the less successful the processing(requesting or otherwise), the worse the cloud will be.

|ISSN 2349-3216| Volume 9, Issue 2, March – April 2026| Bimonthly, Peer Reviewed and Scholarly Indexed Journal|

DOI: 10.15662/IJAESIT.2026.0902004

4.4 Comparative Assessment

A solution is suggested for the adaptive mechanism that is compared to the working solution of the static circuit breaker solution on a circuit breaker. The static method is based on a solid threshold (number of failed requests, time spent or timeout). The proposed framework, however, can dynamically change the thresholds by using real-time anomaly scores and prediction of failure probability in real-time. It can offer a comparison on the benefits of determining whether or not to change the workload situation with the help of AI. The dynamic framework shouldn't trigger breakers in unnecessary situations and needs to be able to notice the failure in risk earlier and in an effective manner take care of overload in the dependent services.

4.5 Expected Evaluation Outcome

As far as the introduced scheme is concerned with the improved pro-active fault management of distributed Cloud platforms, hopefully the evaluation will come out with a positive result. The system is based on continuous monitoring, machine learning based prediction and adaptive circuit breaker control that allows the identification of early warning indicators prior to total service failure. It, therefore, reduces cascading failure, provides fault isolation, reduces downtime and can help with a self healing cloud operation. In the end, the framework assessment verifies the viability of adopting AI helped adaptive circuit breaker observing for a shrewd and extensional solutions on reliability to supplant the rule-based solution.

V. CONCLUSION AND FUTURE WORK

In this research article, an AI-based adaptive circuit breaker monitoring framework and failure prediction for improving reliability for Distributed cloud platform. Distributed cloud architectures are very dynamic—microservices, virtual machines, containers, edge nodes and a large number of network connections and storage solutions come and go on a continuous basis, requiring information propagation to occur continuously. This can lead to cascading failure should one of the services fail, even in such a complex scenario. This can be minimized using traditional circuit breaker mechanisms, but the drawbacks of these circuits is that they're threshold-based so they wouldn't work and are not useful if there are differing workloads and traffic conditions.

The proposed framework overcomes this limitation by incorporating the real-time telemetry monitoring, anomaly detection based on AI, predictive failure modelling, and adaptive circuit breaker control. The framework captures metrics including latency, error rate, throughput, CPU usage, memory usage, network delays, etc., and creates a real-time portrait of systems health and failures of solutions/services that depend on these. The AI layer will produce any abnormal pattern and calculate the probability of service failure occurrence and to detect the danger of any service failure, before the actual service failure. This is an adaptive layer of decisions used to adaptively adjust the circuit-breaker thresholds and switch between closed / open / half-open states of the circuit-breaker. This allows system to quarantine the unhealty services at the optimum time, minimize unnecessary recalcitrant requests, avoid overload and facilitate quicker recovery.

The main novelty of this study is to make an intelligent, context aware and self adaptive circuit breaker which is typically used as an expensive resilience mechanism. These characteristics are supported by the framework: Positive treatment of failure, No downtime - high service availability and possibly self-healing in distributed cloud based applications. It also provides an scalable idea to cope with the uncertainties of clouds: changing workload and changing failure behaviours.

Future works are planned for real life deployment of the proposed approach and integration components, such as deployment in a Kubernettes environment with integration of observability solutions for the deployment. Future research will involve using multi-layered various machine learning and deep learning models (Autoencoder, random forest, hybrid model, etc.) for failure forecasting. Also, the framework can be extended to reinforcement learning and thereby adaptively optimize the circuit breaker decisions. Future work also involves conducting large-scale benchmarking and failure prediction which will provide security awareness, as well as employing explainable AI techniques which will further promote transparency regarding automated decisions and, edge-cloud deployments

REFERENCES

[1] X. Zhou et al., “Fault analysis and debugging of microservice systems: Industrial survey, benchmark system, and empirical study,” IEEE Transactions on Software Engineering, vol. 47, no. 2, pp. 243–260, Feb. 2021.

[2] M. Ganesan, “Transforming home electronics customer self-installation experience with AI,” International Journal of Research Publications in Engineering, Technology and Management (IJRPETM), vol. 7, no. 4, pp. 14319–14327, 2024.

|ISSN 2349-3216| Volume 9, Issue 2, March – April 2026| Bimonthly, Peer Reviewed and Scholarly Indexed Journal|

DOI: 10.15662/IJAESIT.2026.0902004

[3] N. Zhao et al., "Identifying bad software changes via multimodal anomaly detection for online service systems," in Proc. 29th ACM Joint Meeting European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Aug. 2021, pp. 527–539.
[4] P. Chen, Y. Qi, and D. Hou, "CauseInfer: Automated end-to-end performance diagnosis with hierarchical causality graph in cloud environment," IEEE Transactions on Services Computing, vol. 12, no. 2, pp. 214–230, Mar. 2019.
[5] M. Jin et al., "An anomaly detection algorithm for microservice architecture based on robust principal component analysis," IEEE Access, vol. 8, pp. 226397–226408, 2020.
[6] T. Wang, W. Zhang, J. Xu, and Z. Gu, "Workflow-aware automatic fault diagnosis for microservice-based applications with statistics," IEEE Transactions on Network and Service Management, vol. 17, no. 4, pp. 2350–2363, Dec. 2020.
[7] M. Raeiszadeh, A. Ebrahimzadeh, A. Saleem, R. H. Glitho, J. Eker, and R. A. F. Mini, "Real-time anomaly detection using distributed tracing in microservice cloud applications," in Proc. IEEE 12th International Conference on Cloud Networking (CloudNet), Nov. 2023, pp. 36–44.
[8] Y. Gan et al., "Seer: Leveraging big data to navigate the complexity of performance debugging in cloud microservices," in Proc. ACM International Conference on Architectural Support for Programming Languages and Operating Systems, 2019, pp. 19–33.
[9] S. Nedelkoski, J. Cardoso, and O. Kao, "Anomaly detection and classification using distributed tracing and deep learning," in Proc. 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID), May 2019, pp. 241–250.
[10] J. Bogatinovski, S. Nedelkoski, J. Cardoso, and O. Kao, "Self-supervised anomaly detection from distributed traces," in Proc. IEEE/ACM 13th International Conference on Utility and Cloud Computing (UCC), Dec. 2020, pp. 342–347.
[11] P. Liu et al., "Unsupervised detection of microservice trace anomalies through service-level deep Bayesian networks," in Proc. IEEE 31st International Symposium on Software Reliability Engineering (ISSRE), Oct. 2020, pp. 48–58.
[12] L. Meng, F. Ji, Y. Sun, and T. Wang, "Detecting anomalies in microservices with execution trace comparison," Future Generation Computer Systems, vol. 116, pp. 291–301, Mar. 2021.